Quantifying New Zealand's fault lines: fractal dimension and orientation

Claudia Seibold
Department of Mathematics and Statistics
University of Canterbury

This thesis is submitted for the degree of
MSc in Statistics

2014

Acknowlegements

This work would not have been possible without the support, encouragement, patience, academic experience and assistance of many people. Firstly, I would like to thank my supervisors Prof Jennifer Brown and Dr Elena Moltchanova for their for their support and guidance throughout my study. I would like to thank Prof Tim Davies at the Department of Geological Sciences for his suggestions and recommendations. Further, I would like to acknowledge GNS Science, a crown research institute in New Zealand specializing in Earth systems. GNS supplied the field data for this study as part of the geological map data QMAP Seamless GIS 2012 pre-release version. Special thanks go Steve Gourdie and Paul Brouwers for assisting me with all computer-related issues. Finally, to my friends, thank you all for your unconditional love and support, for being with me during the dark times and celebrating the good times together.

Abstract

The surface of New Zealand is dissected by a complex network of fault lines. Understanding the mathematics behind this pattern could lead to better prediction of earthquakes, both large and small, that shake the country on an average of 14,000 times a year. The object of this thesis is to investigate whether fault lines are fractal and, if so, what their fractal dimension is. A spatial pattern of fault line orientation is studied, using a permutation test, specifically adapted for this study. It is found that the estimated fractal dimension is very dependent on the method chosen to analyse spatial distribution. The characteristic of the estimated fractal pattern for the South Island of New Zealand as described in this study and appears to agree with findings for the seismic area of Japan. The orientation of the fault lines depends on spatial location. This result supports leading theories of plate tectonics. This study helps build an overall picture of the fault system in New Zealand and brings us one step further towards a more complete knowledge which may one day help us to predict and prepare for earthquakes in the future.

Contents

Notation

Chapter 3 - 5

Ed	Euclidean dimension
td	topological dimension
ed	embedded dimension
s	scaling factor
e	box edge length
m	magnification factor
m_O	original magnification factor
m_A	adjusted magnification factor
N	number of occupied boxes
l_R	interpolation length - relative to box edge length
l_U	interpolation length - in units
X_r and Y_r	rotated X and Y coordinate values
X_t and Y_t	trimmed X and Y coordinate values
X_l and Y_l	trimmed interpolated X and Y coordinate values
X_h and Y_v	trimmed horizontally translated X and vertically translated Y coordinate values
X_d and Y_d	trimmed rotated X and Y coordinate values
h and v	horizontal and vertical translation
h_R and v_R	horizontal and vertical translation - step relative to box edge length
h_U and v_U	horizontal and vertical translation - step in the units of metre
fd	fractal dimension
fd_S	similarity fractal dimension
fd_{WD}	walking divider fractal dimension
fd_{LR}	box-counting local fractal dimension calculated as log ratio
$fd_{LR(m)}$	box-counting local fractal dimension calculated as log ratio for horizontal at magnification factor m
$fd_{LR(m,\ h_r,\ v_r)}$	box-counting local fractal dimension calculated as log ratio for horizontal and vertical translated coordinate values
$fd_{LR(m,\ h_r,\ v_r)}$	box-counting local fractal dimension calculated as log ratio for horizontal and vertical translated coordinate values
$fd_{LR(m,\ d)}$	box-counting local fractal dimension calculated as log ratio for rotated coordinate values
fd_{FD}	box-counting local fractal dimension calculated with forward differences
$fd_{FD(m_1,\ m_2)}$	box-counting local fractal dimension calculated with forward differences between magnification factors m_1 and m_2
fd_{MD}	box-counting local fractal dimension calculated with mixed differences
fd_{GM}	box-counting global fractal dimension calculated as mean of fd_{LR}
fd_{GS}	box-counting global fractal dimension calculated as slope of a line fitted to $log(m)$ and $log(N)$

Chapter 6

$D_{2KS(m_1,\ m_2)}$	test statistic for Kolmogorov-Smirnov test for the groups m_1 and m_2
$D_{MKS(m_1,\ m_2,\ m_3)}$	test statistic for Kolmogorov-Smirnov test for the groups m_1, m_2 and m_3

Chapter 1

Introduction

The forces of Nature cause radical changes as they have the power to shape the Earth. People benefit or suffer from the consequences. From one moment to the next, an earthquake or a volcanic eruption can abruptly release energy in magnitudes far greater than anything humans can produce. This abrupt trigger of eruptions and earthquakes stands in contrast to the immensely long time spans over which the underlying Earth's processes operate. While humans have no control over processes that occur in the interior of our planet and the related forces that cause such events, we have the opportunity to study them. The more we understand about these interior processes, the better we can treasure the beauty created by them, as well as prepare for and deal with their negative effects.

This thesis focuses on quantifying the fault system in the South Island of New Zealand, and in particular the spatial distribution and orientation of the fault lines. Research into the fault systems is very relevant to the study of earthquakes and has become a hot topic in New Zealand, especially after the Canterbury earthquakes that started in 2010, and still continue today. Tectonic processes began to attract my interest when I first visited the exhibition *Awesome Forces*, a display of the geological forces that shape New Zealand's landscape, presented at Te Papa museum in Wellington. Seeing the impact of these forces on New Zealand's landscape while travelling around the South Island has motivated me to study this field and to contribute to its development.

Chapter 2 begins by describing the contextual geology and the reasons why tectonic plates move and build up pressure in certain areas. This kind of movement produces different types of plate boundaries and faults. The fault system of New Zealand is described against this theoretical background and an argument is made for the adequacy of the fractal analysis of spatial distribution of fault lines.

Chapter 3 - 5 focus on identifying the fractal dimension for the spatial distribution of fault lines.

Chapter 3 presents fundamentals of fractal theory and defines the different types of fractal dimensions. The use of the *Box-Counting* method for this study is explained. Numerous examples of the application of fractal analyses of this method in medicine, engineering and geology are provided.

Chapter 4 describes in detail the Box-Counting method and relevant mathematical procedures. They are demonstrated with an artificial fractal pattern known as the Koch Curve. Estimates of the fractal dimension are produced and compared between different constructions. User choices for the Box-Counting method are considered. By comparing the resulting estimated fractal dimension against the known fractal dimension of the Koch Curve, the implications of these different choices can be understood. This then allows the user to select values of relevance when applying the Box-Counting method to the fault lines of interest.

Chapter 5 estimates the fractal dimension of the spatial distribution of fault lines. This is done with improved methods and awareness of the critical choices in Box-Counting gained from Chapter 4. The dataset consists of fault lines with a total length of more than 11,000 km covering an area of 96 km $\times$ 324 km. Extreme differences in the estimated fractal dimension are discussed and compared to results from a similar study in Japan.

Chapter 6 focuses on identifying differences in the orientation of the fault lines between geographic areas.

The difference in the orientation of fault lines between several geographic areas on the South Island of New

Zealand is tested statistically. Generally, the Kolmogorov-Smirnov test for two groups (2KS) is used to compare two areas, however, in this study, multiple areas are evaluated. To this end, a permutation Kolmogorov-Smirnov test for multiple groups (MKS) has been developed. This adjusted test provides simultaneous comparisons of several areas while keeping the type 1 error constant. The combined application of the 2KS and MKS provides the opportunity to test whether there is any statistically significant difference between any areas, and then to locate where this difference lies.

Chapter 7 provides a short summary of the thesis and proposes some avenues of future research based on the findings presented here.

Chapter 2

Contextual Geology

This chapter gives an overview of historical and leading theories of plate tectonics. The fault system in the South Island of New Zealand is explained against this geological background.

2.1 Continental Drift

The theory of continental drift was proposed by the German scientist Alfred Wegener in 1915. In his book *The Origin of Continents and Oceans*[107] he suggested that the Earth's continents had once been joined in one large continent. He named this continent *Urkontinent* which means primary continent and is equivalent to the Greek *Pangea*. About 250 million year ago Pangea split into a northern and southern landmass[94]. Christiansen[20] explains that the northern part of Pangea, known as Laurasia, formed Europe, Asia, North America, and Greenland. The southern supercontinent consisted of present Africa, South America, India, Madagascar, Australia and New Zealand. The Austrian geologist Eduard Suess[29] called this southern part Gondwana in reference to formations in the Era of upper Paleozoic and Mesozoic about 2500 to 1000 Ma (million of years ago) in a region called Gondwana in central India. His reason was that the formations in India are similar to those on the Southern continents. Wegener[107] reconstructed the Earth's map based on the movement of continents for three different geological ages shown in Figure 2.1. His theory for this movement was that continents are floating on the underlying layer of the Earth like ice sheets on the sea. This theory is confirmed by the geological relationship between the continents, as well as fauna and flora from past geological eras.

New Zealand's short but unique history can be interpreted in Wegener's theory beginning 545 million years ago. This theory is based on a geologic time scale, a system of chronological measurement that relates stratigraphy to time. This system helps to describe the age and relationships between events in Earth's history. Hamblin & Christiansen[44] compare the length of geologic time to a football field of the length 100 yards (91.44 metres). The whole length of the football field in Figure 2.2 represents the Earth's age of about 4,000 Ma (million years from the present), dinosaurs first appeared five yards from the goal line and the historic time cannot be represented as it is too short. On this football field scale New Zealand's development starts in the last thirteenth yard. The development can be divided into sedimentation taking place from 545 to 142 Ma and mountain building, or orogeny, since 142 Ma to the modern age[97]. In the time of sedimentation and even in the beginning of mountain building, New Zealand was still part of Gondwana.

Mortimer[71] describes that between 545 Ma and 99 Ma, New Zealand consisted of nine major volcano-sedimentary terranes, three regional batholiths, and three regional metamorphic-tectonic belts that overprint the terranes and batholiths shown in Figure 2.3. Hamblin & Christiansen[44] explain that batholiths are masses of rock

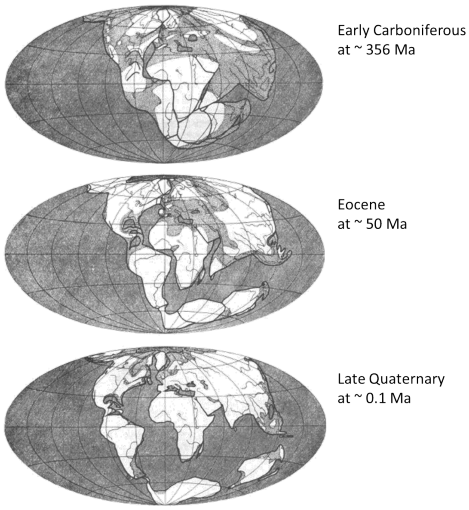


Figure 2.1 ***Wegener's[107] visual representation of the breakup of Gondwana.*** *Wegener[107] reconstructed the Earth's map based on the movement of continents for three different geological ages: Early Carboniferous at about 356 Ma, Eocene at about 50 Ma, and Late Quaternary about 0.1 Ma (million of years ago).*

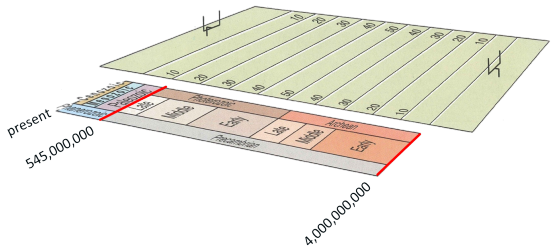


Figure 2.2 ***Football field representing the Earth's age.*** *The whole length of 100 yards (91.44 metres) of the football field represents the Earth's age of about 4,000 Ma (million years from the present). The numbers 10, 20, ..., 50 on the football field indicate the length of the field in yards starting from both sides and ending in the center. Dinosaurs first appeared five yards from the goal line and the present historic time cannot be represented as it is too short. On this football field scale, New Zealand's development starts in the last thirteenth yard around 545 Ma.*
(figure from Hamblin & Christiansen[44])

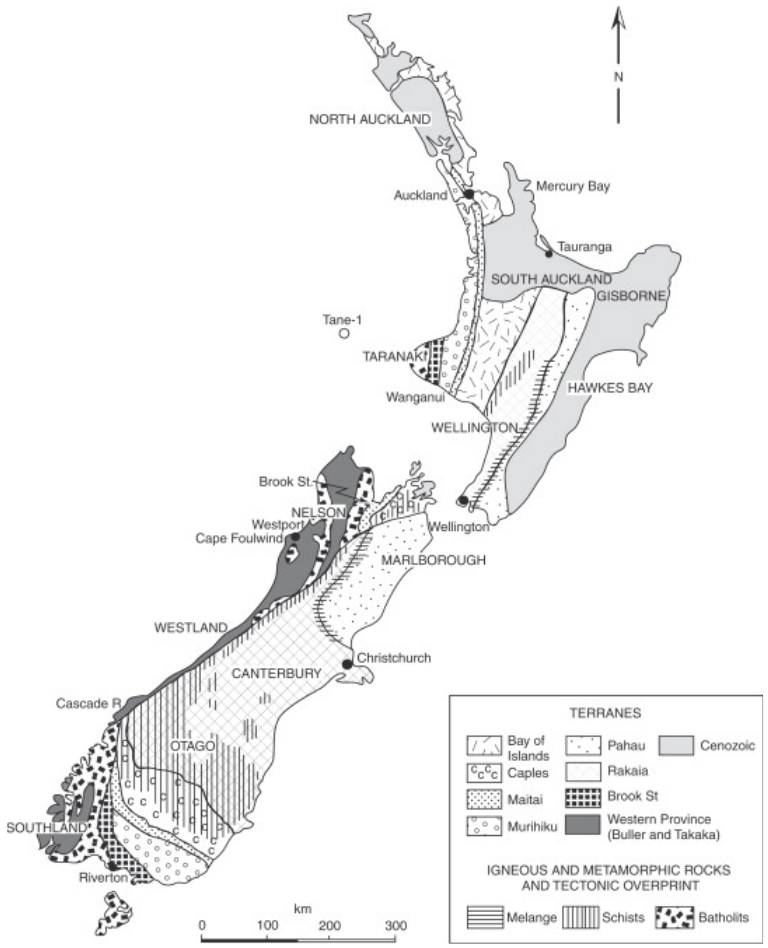


Figure 2.3 ***The use of rock types in understanding New Zealand's geological history.*** *The Identification of rock types in New Zealand is used to assign events to the Geological Time Scale. This helps to understand relationships between events in the Earth's history. In the time 545 Ma (million years ago) to 99 Ma, New Zealand existed as a compilation of nine major volcano-sedimentary terranes, three regional batholiths, and three regional metamorphic-tectonic belts that overprint the terranes and batholiths.*
(from Smale[91] after Mortimer[71])

Figure 2.4 ***New Zealand's position on the border between the Australian and Pacific plate.*** *The arrows between the Australian and the Antarctic plate indicate a ridge where the Australian plate moves away from the Antarctic plate, and builds up stress against the Pacific plate. This movement is one reason for New Zealand's daily earthquakes.*
(from Geographical Association[34])

larger than 100 km^2 formed when magma cools beneath the surface. While the batholiths of New Zealand were created by fluid magma, the metamorphic rock of New Zealand was formed by pre-existing rock in solid state. Metamorphic-tectonic belts were created through the pressure and change in temperature while two tectonic plates were moving towards each other.

Around 99 Ma, New Zealand broke away from Gondwana[97]. The movement of New Zealand resulted in the gradual formation of the Tasman Sea between New Zealand and Australia. About 35 Ma New Zealand stopped moving away from Australia and begun to sink. At that time a new rift formed between Australia and Antarctica resulting in a movement of Australia and build up of strain in the southwest Pacific crust shown with red arrows in Figure 2.4. The pressure of this movement has not stopped since uplifting of central Westland of New Zealand producing the Southern Alps, but still continues to this day[97].

2.2 Plate Tectonics

Wegener's continental drift theory stating that continents are floating on the underlying fluid Earth's layer like ice sheets on the sea was improved by the theory of plate tectonics. Geologists use the term *plate* for one of the very large pieces of rock that form the Earth's surface and move slowly. The term *tectonic* derives from the Greek word *tekton*, meaning carpenter or builder. These words together refer to the building of the Earth's surface by the moving plates.

The theory of plate tectonics states that the Earth's skin is fragmented into ten or more tectonic plates. Looking at the inner Earth layers, the plates compose the lithosphere which is the crust and uppermost solid mantle. For an easier understanding of the thickness of the lithosphere, a peach can be used as a model for the Earth[45;66]. The peel is compared with the Earth's crust and the flesh with the mantle. The lithosphere would be the peel plus a thin outermost layer of flesh visualized in Figure 2.5. New Zealand is on the border between two of these big tectonic plates, the Australian and Pacific plate shown in Figure 2.4. The Australian plate is much bigger than the landmass of Australia.

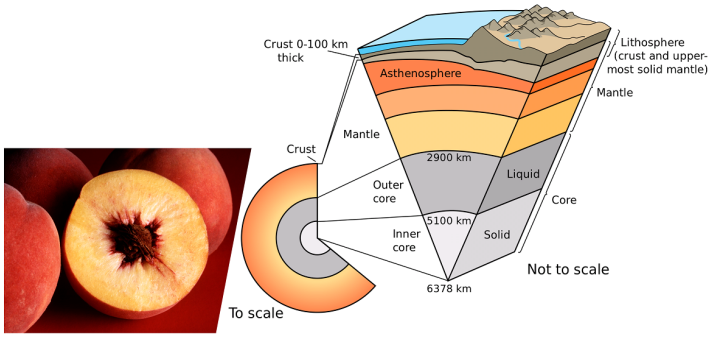


Figure 2.5 ***Peach model of the inner Earth.*** *Using a peach as a model of the Earth, the peel and the outermost flesh represent the Earth's lithosphere whose fragments are the tectonic plates.*
(illustration on the right from USGS[101])

In 1928, Arthur Holmes[78] proposed a continental drift through subcrustal thermal convection currents in Earth's mantle. The convection currents rise, spread below a plate boundary and converge and descend along another one illustrated in Figure 2.6. The plates move in the same direction as the convection. The driving forces of convection are only one explanation, in contrast, Kious & Tilling[50] state that magma is rising and the extension pushes plates apart thereby causing plate movement. Marshak[64] casts this magma pushing theory into question and explains that the magma is not pushing the plates away but is pulled by the moving plates. The proposed reason is that the surface of the sea floor slopes away from an underwater mountain range, called a ridge, and gravity pushes on the sea floor that lies farther from the ridge. When the outermost layer of the Earth moves away

from the ridge, it creates a gap, and rising magma fills this gap. These different theories demonstrate that the movement of plates across the Earth's surface is not yet understood.

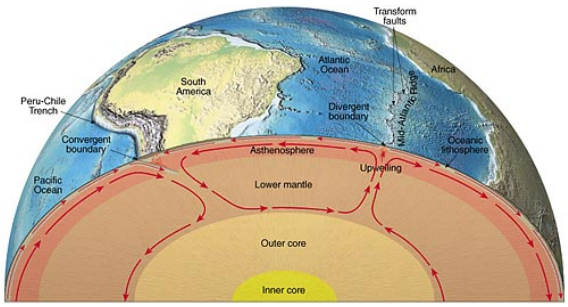


Figure 2.6 ***Convection along plate boundaries.*** *The convection currents rise and spread below a plate boundary, moving the tectonic plate to converge and descend along another boundary.*
(from Redwood[21])

2.3 Plate Boundaries

The way two tectonic plates interact with each other characterizes the boundary between them producing a specific pattern of earthquakes, volcanism, and topography. The three major ways of interaction are moving away from each other at a divergent boundary, moving toward each other at a convergent boundary and passing each other at a transform boundary illustrated in Figure 2.7.

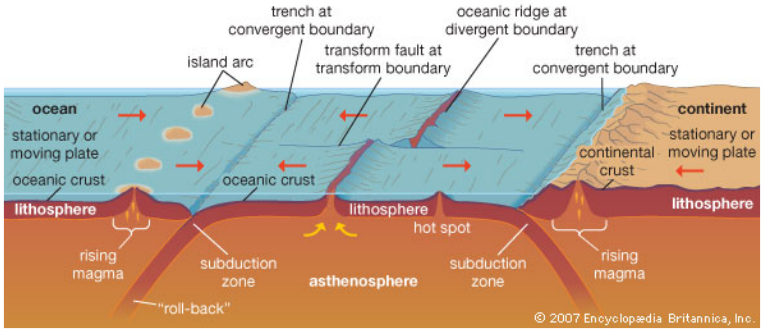


Figure 2.7 ***Three major ways tectonic plates interact with each other.*** *At a divergent plate boundary the tectonic plates move away from each other. At a convergent boundary they move toward each other, and at a transform boundary they pass each other.*
(from Encyclopedia Britannica[30])

Christiansen[20] states that between plates at a divergent boundary, magma, or molten rock, can come to the surface and erupt from a volcano as lava. When cooling down, this lava creates new parts of the Earth's crust. The tensional stresses of the plates can produce long rift zones and normal type faults (see Section 2.4). In New Zealand there is no divergent plate boundary. The Pacific and the Australian Plate do not move away from each other, but rather towards each other and break up crust. The direction of the movement is not perpendicular but slightly skewed. Therefore the pressure between the two plates is not constant. The change in the orientation is visible at the Alpine fault in Figure 2.8

The convergence looks different depending on the categories of crusts interacting at a boundary. The categories are continental and oceanic crust. Moores & Twiss[70] describe continental crust as thicker and having a more complex structure because it is older and has experienced a longer tectonic history than oceanic crust. Christiansen[20] explains that the oceanic crust is denser (about 3.0 g/cm^3) than the continental crust (about 2.8 g/cm^3). Most plates

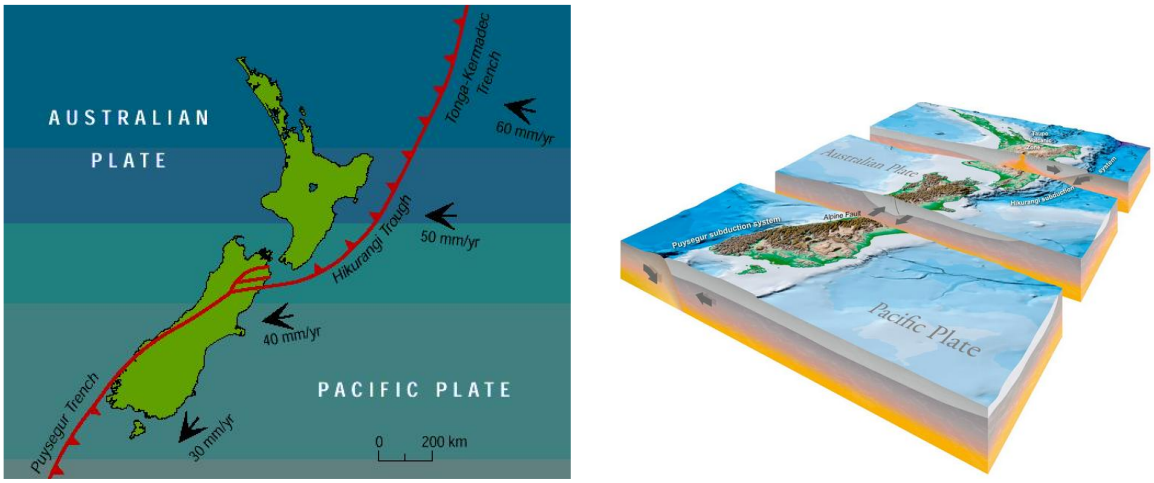


Figure 2.8 ***The curve of the Alpine Fault in New Zealand coupling trenches.***
left: *In New Zealand, the Australian and Pacific Plates collide along a curving boundary. The way they interact with each other changes along the boundary. The Alpine Fault is a transform plate boundary coupling the Tonga-Kermadec Trench in the North and the Puysegur Trench in the South. Both trenches are convergent plate boundaries.*
left and right: *In the North the Pacific Plate is dipping under the Australian Plate (indicated by the red arrows in the left figure, and in grey arrows in the right figure), in the South it is the other way around as indicated by the arrows. The Alpine fault transforms from one direction to the other.*
right: *In the middle, through most of the South Island the two plates pass each other along the Alpine Fault as indicated by the grey arrows.*
(left figure from Reyner[85], right figure from GNS Science[40])

are comprised of both continental and oceanic crust, but the giant Pacific Plate that partly contains New Zealand is almost entirely oceanic. If both converging plates contain oceanic crust, one is subducted into the mantle. The same happens if oceanic meets continental crust. It is always the oceanic crust that is pushed below as it is heavier than the continental crust. Kious & Tilling[51] state that if two continental crust plates meet head-on, neither is subducted because the continental crust is too buoyant and resists downward motion. The mass in the collision zone becomes compressed, and the pressure creates folded mountains. It is not only the density of the plate but also the convergence rate, the direction and the thermal structure of the subduction zone that finally characterize a convergent plate boundary. New Zealand is situated at the boundary of two convergent plates.

Lewis, Scott & Nodder[58] explain that at the Puysegur Trench in the south-west of New Zealand, the oceanic crust of the Australian Plate dives beneath the Pacific Plate as indicated with arrows in Figure 2.8. The Kermadec-Tonga Trench between the North Island of New Zealand and the island of Tonga is a mirror image of the Puysegur Trench. It is a subduction zone, where the older oceanic crust of the Pacific Plate is sinking below the more buoyant, young oceanic crust of the Australian Plate[76].

The Alpine Fault connects both of these convergent plate boundaries, the Puysegur Trench and the Kermadec-Tonga Trench. The Alpine fault is categorized as a *transform* plate boundary. Hamblin & Christiansen[44] use the term transform to refer to the way these boundaries transform one type of plate motion into another type. In contrast to divergent and convergent plate boundaries, a transform plate boundary neither produces nor breaks up the crust. At a transform boundary, the plates pass each other horizontally, illustrated with grey arrows in Figure 2.8 (right). There are three possibilities depending on whether the transform plate boundary connects ridges or trenches, or both. A ridge-ridge transform plate boundary is between two divergent plate boundaries, a ridge-trench boundary transforms from a divergent to a convergent plate boundary, and trench-trench boundary couples trenches at two different convergent plate boundaries. Hamblin & Christiansen[44] state that while ridge-ridge boundaries are very common, ridge-trench and trench-trench boundaries are rare. The Alpine Fault is one of these rare trench-trench transform faults shown in Figure 2.8 (left). Egger[26] highlights the inland position of the Alpine fault as noteworthy because most transform boundaries occur not on land but along mid-ocean ridges. Earthquakes at transform faults are shallower than at divergent and convergent plate boundaries. This characteristic of the Alpine fault combined with its inland position means that it has the potential to significantly affect people through earthquakes.

2.4 Fault Lines

A fault line is a fracture along which the crust has moved. It can be, but need not be along a plate boundary. Fossen[31] defines a fault as any surface or narrow zone with visible shear displacement along the zone.

Faults are geometrically grouped in dip-slip, strike-slip and oblique-slip faults. The movement of rock at dip-slip faults is either down or up along the dip direction illustrated in Figure 2.9. When two blocks of rock are pulled apart, and the hanging wall has moved downward relative to the footwall, the fault is defined as *normal*. In the opposite case, when the forces press the two blocks of rocks together and the rock above the fault moves up, the fault is defined as *reverse*. Reverse faults with an angle $< 45^\circ$ are called *thrust*. Divergent plate boundaries are normal, and convergent boundaries are reverse faults. In contrast, the movement of rocks at transform plate boundaries is termed *strike-slip*. The walls of strike-slip faults move not up or down but sideways. The forces move the plates either left-lateral or right-lateral. Right-lateral means that if you were standing near the fault line and looking across it, you would see the corresponding rock on the other side more to the right. The Greendale fault on the South Island of New Zealand is such a right-lateral strike-slip fault as you can see from the movement of the hedge in Figure 2.10. In reality, many fault lines do not show these pure geometrical characteristics but a combination of dip-slip and strike-slip called oblique-slip. Previous study provide evidence which indicates that the Darfield Earthquake in 2010 in New Zealand was a complex event involving both compressional thrust and shear strike-slip motion (Campbell[19]).

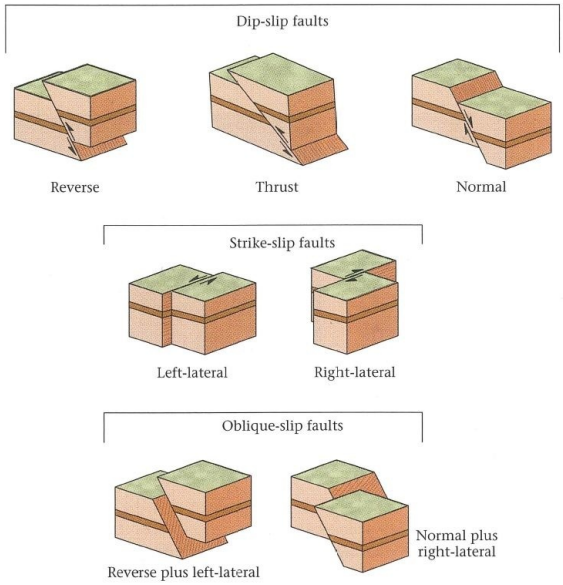


Figure 2.9 ***Geometrical grouping of faults.*** *The direction of dip-slip faults is up or down depending whether the tectonic plates move towards or away from each other. When the tectonic plates pass each other sideways, the fault is termed strike-slip. Many fault lines are a combination of dip-slip and strike-slip called oblique-slip.*
(from Marshak[64])

Figure 2.10 ***Lateral strike-slip Greendale fault.*** *The movement of the hedge at the September 2010 earthquake in the Canterbury region indicates that the Greendale fault in the South Island of New Zealand is a right lateral strike-slip fault.*
(from GNS[39]

2.5 Earthquakes in New Zealand

In New Zealand earthquakes happen daily as illustrated in Figure 2.11. Oldershaw[77] explains that most earthquakes happen at the edges of plates that make up the Earth's crust. New Zealand is placed on such a plate boundary between the Australian and the Pacific plate. As the plates collide at convergent plate boundaries or grind past one another at transform boundaries, stress accumulates locally and is released during earthquakes. The movement of the Australian and Pacific plates towards each other resulted in the uplifting of Westland in New Zealand and producing the Southern Alps. This pressure still continues and causes earthquakes everyday.

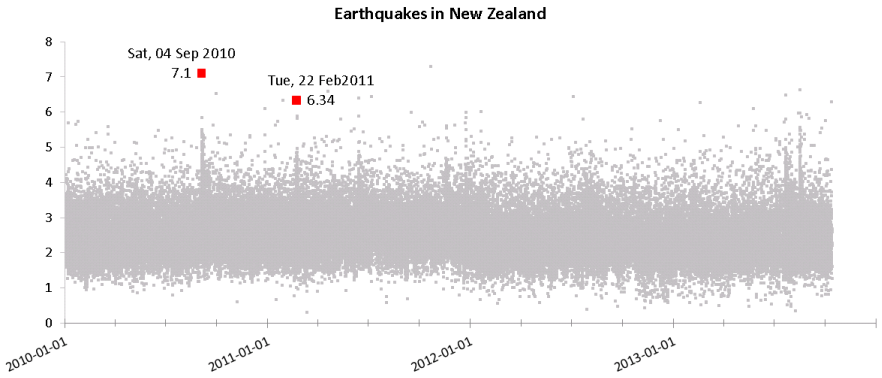
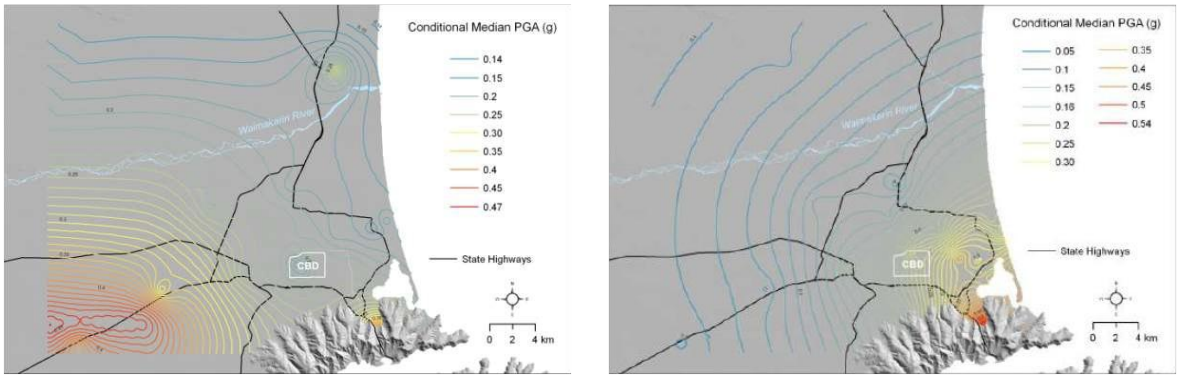


Figure 2.11 ***Daily earthquakes in New Zealand.***

Most of these earthquakes are too small or too far away from human settlements to cause significant damage. However, if an earthquake is close and shallow, the consequences are dramatic as displayed in the contrast between the 04 September 2010 and the 22 February 2011 earthquake close to Christchurch, the second highest populated city in New Zealand.
(data download from GeoNet[35])

Not all of these earthquakes cause damage. The magnitude of most earthquakes is small, or they occur in areas without settlement. Therefore the probability of an earthquake that critically affects people is low, however, the consequences of such earthquakes are high. In 2010 and 2011 two major earthquakes struck in the Canterbury region on the South Island of New Zealand. On 4 September 2010 a 7.1 magnitude earthquake revealed a 29 km long fault break, where no active faults had previously been mapped. This fault has now been named the Greendale Fault. The northern side of the fault moved about 4 metres to the east relative to the southern side, indicating an east-striking transform fault and illustrated in Figure 2.10. The earthquake's epicentre was located about 40 km west of Christchurch close to Darfield, thus the name of this event. The epicentre is the point on the Earth's surface that is directly above the location where an earthquake originates. The actual location of the earthquake within the Earth is called the hypocentre. For the Darfield Earthquake the hypocentre was about 10 km deep. Both the parameters geographical location and depth are important. The closer to settlement and the more shallow an earthquake occurs, the more extreme the damage. This is one of the reasons why the smaller 6.3 earthquake on 22 February 2011 was more damaging and resulted in 185 deaths. The earthquake was centered 10 km south-east of central Christchurch at 6 km depth[74]. The high damage is explained by large peak ground acceleration in and around the Central Business District (CBD) visualized by Bradley & Hughes[18] with a white border in Figure 2.12. This ground motion resulted in extensive damage to lifelines, residential, commercial and industrial structures. New Zealand police[75] stated that the rebuilding costs of Christchurch and the surrounding Canterbury region were estimated at 40 billion NZD in April 2013.



September 2010 *February 2011*

Figure 2.12 ***Ground motion in Christchurch during the September 2010 and February 2011 Canterbury earthquakes.*** *The spatial distribution of ground motion shows that the Central Business District of Christchurch suffered higher peak ground acceleration in the earthquake in February 2011 (right) than in September 2010 (left). The higher value in February 2011 resulted in much more extreme structural damage and loss of life than the September 2010 earthquake.*
(from Bradley & Hughes[18])

2.6 Euclidean, Granular and Fractal View of Faults

In the past the geometrical, mechanical and mathematical nature (GMM) of faults was mainly analyzed on the basis of three very different views: the Euclidean, the granular and the fractal view.

The Euclidean view claims that the GMM structure of faults is fundamentally smooth and continuous, whereas the fractal view argues that faults like many other patterns of nature are so irregular and fragmented, that, compared to standard Euclidean geometry, they must be more complex (Ben-Zion & Sammis[17]). The complex pattern of the fault network of the South Island of New Zealand displayed in Figure 2.13 indicates that the fractal view may be more appropriate.

The granular view focuses on the local body of faults while the fractal view describes the fault network more globally. This study focuses on the spatial network rather than the characteristic of a single fault, and this network appears to be irregular rather than smooth. For this reason the fractal framework is used to analyze the fault system on the South Island.

2.7 Fractals

Faults can be described as complex fractal structures. Cowie, Vanneste & Sornette[23] point out that these structures are developed by ruptures correlated in space and time. Fractal analysis involves examination of an overall fault system pattern and identifying similar patterns in a smaller size but having exactly the same shape. Results range from no fractal pattern at all, one fractal pattern for the complete fault system, or two or more fractal patterns for different scales. The latter possibility has also been termed band-limited fractal structure, and Lei and Kusunose[57] have found empirical support for this in Japan. This study used Box-Counting and found a break in the estimated fractal dimension of rivers, faults and earthquakes shown in Figure 2.14. This break is at 13 kilometers which means there are two different fractal patterns, one for boxes smaller than 13km and one for boxes bigger than 13km. Kato & Lei[48] suggest that these breaks in the fractal structure might be explained by the characteristic length of faults which is controlled by the strain rate, the viscoelastic relaxation time and the thickness of the elastic layer.

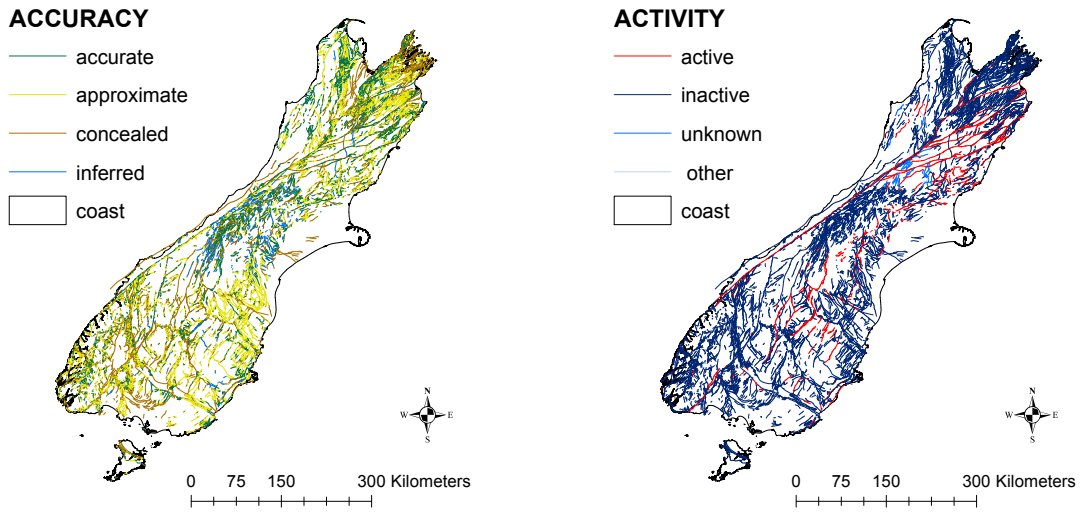


Figure 2.13 ***Fault lines of QMAP Seamless GIS 2012 pre-release version.*** *More than 27,000 fault lines, summing up to more than 43,000 km, are part of the geological map data QMAP Seamless GIS 2012 pre-release version*[41]*.*
There are many more accurate, approximate and concealed fault lines than inferred fault lines:
7889 accurate, 7836 approximate, 8179 concealed and 802 inferred polylines.
Most of the fault lines are inactive: 3,205 active, 21,164 inactive, 318 unknown and 19 not categorized.

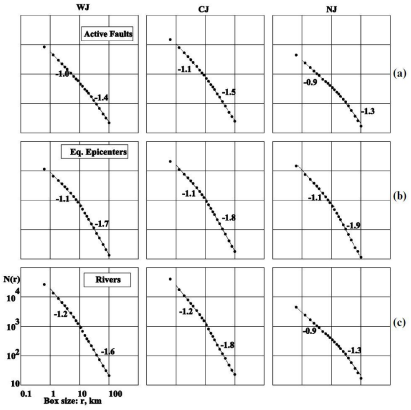


Figure 2.14 ***Japan's Box-Counting fractal dimension for earthquake epicentres, active fault and rivers.*** *Lei & Kusunose*[57] *used Box-Counting for active faults, earthquake epicentres, and rivers, in the three areas north-east Japan (NJ), central Japan (CJ), and west Japan (WJ). The common feature of these results is that the slope of the regression lines break at 13km. This break divides the range into two bands and therefore this fractals are called band-limited. Note, they used square boxes, and therefore could label the x-axis with the box edge length. This results in a negative slope. In Chapter 4 and 5, square and rectangular boxes are used. For the purpose of comparison between these box shapes, the x-axis is labelled with the magnification factor. This results in a positive slope.*
(from Lei and Kusunove[57]*)*

2.8 Self-organized criticality (SOC)

The fractal framework of fault lines implies self-organized criticality of earthquakes. Ben-Zion et al.[17] propose that the fractal pattern reflects a dynamic balance between two opposing tendencies - strain weakening and strain hardening processes illustrated in Figure 2.15. Fossen[31] explains that for strain hardening to occur, the stress necessary to deform the rock must be increased for pressure to accumulate, because the rock becomes stronger and harder to deform. This is the basis of the granular view. The opposite, strain weakening is the effect in which the stress level must be decreased in order to maintain a fixed pressure rate, and the deformation processes and structures are expected to evolve toward the continuum-Euclidean framework.

Turcotte[98] points out that the balance between hardening and softening is critically tuned to produce neither positive nor negative overall feedback during deformation. *Critically* in this context means that the balance never reaches the hypothetical critical state. Vallianatos, Kokinou & Sammonds[102] emphasize that fault systems are among the most relevant paradigms of this balancing process, which results in a complex spatial-temporal phenomenon related to deformation and sudden rupture of the Earth's crust. An earthquake is caused when the stress, caused by the relative displacement between two surface plates, approaches the critical value. Turcotte[98] notes that in the event of an earthquake the stress is either transferred to some other area or is released by producing a new fault or deforming an existing one. Therefore the self-organized criticality of earthquakes influences the spatial pattern of faults.

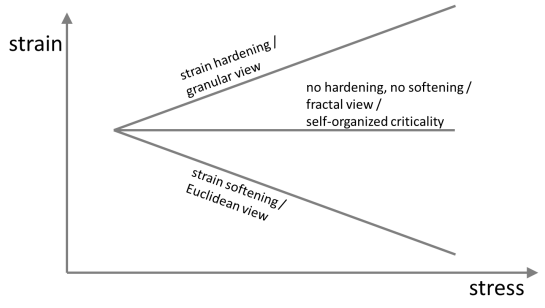


Figure 2.15 ***Stress-strain curve for elastic-plastic materials.*** *For strain hardening the stress necessary to deform the rock must be increased, because the rock becomes stronger and harder to deform. This is expected in the granular view. The opposite, strain softening is where the stress level must be decreased in order to maintain a fixed pressure rate, and the deformation processes and structures are expected to evolve toward the continuum-Euclidean framework. The fractal pattern claims to be a dynamic balance between these two opposing tendencies. The critical tuning between these tendencies is known as self-organized criticality.*
(modified from Fossen[31])

2.9 Connection between fractals, self-organized criticality and power law

Descriptions of fault networks in terms of fractals or self-organized criticality are not mutually exclusive - both are related to power laws. Gloaguen, Marpu and Niemeyer[38] describe fractals using power-law processes, and interpret the exponent as the fractal dimension. Yang[109] suggests that such power-law distributions in self-organized criticality are caused by balancing contradicting processes. For the fault lines the contradictory processes are strain hardening and strain weakening. Gospodinov, Marekova and Marinov[42] state that if the exponent of the power law is an integer, then it is equivalent to a Euclidean dimension (zero for a point, one for a line, two for a square and three for a cube).

2.10 Methods for Structural Analysis

The most direct and important source of information on fault system developments are field observations. Observing which layer of the rock is cut by a fault supports to date the age of the fault, and to arrange events of Earth's history in chronological order. For example, faults are younger then the rock they cut. In contrast to direct measurement, indirect observation is more economical. Remote surveillance methods such as satellite and seismic surveying are becoming increasingly important, as these deliver very accurate digital data. Such digital data is arguably more objective than sketches from field studies. Besides analyzing actual field data, geologists also perform controlled laboratory experiments with artificial data such as sand boxes. The effect, for example the movement of the sand under experimental conditions, such as the force and directional pressure on the sand, can be directly observed. The easy handling is a big advantage compared to real world observations. However, it is not always easy to scale down the physical proportions and properties for a realistic reproduction. With the development in increasing speed of computing, numerical modeling of geological processes becomes simpler. Nevertheless, the processes are complex and even the fastest computer may not be good enough to model this complexity. This study aims to provide numerical characterisation for the fault system in the South Island of New Zealand.

2.11 Geological Data for this Study

GNS Science, a research institute in New Zealand specializing in Earth systems, supplied the field data as part of the geological map data *QMAP Seamless GIS 2012 pre-release version*[41]. QMAP is the new 1:250,000 national geological mapping project. These fault lines can be described either with a geographic or a projected coordinate system. The geographic coordinate system is based on a reference ellipsoid which is a mathematical approximation of the Earth's shape. The specifications of the observed fault lines' positions in the QMAP are based on the geological coordinate system *New Zealand Geodetic Datum 1949 (NZGD1949)*. This system was defined using astronomical observations to establish the coordinates of a relatively small number of locations[89]. Triangulation was then used to connect these places with reference to each other. NZGD1949 is a static datum which means that the coordinates of its locations are fixed with the position at 1949. Deformations due to tectonic processes in the last seventy years have had a deleterious effect on this system. Even if the same astronomical method was applied the coordinates of locations in 1949 would be different from 2013. Coordinates of locations between the 1949 and 2013 differ by about 2 meters. For this study, however, this discrepancy is not obstructive. A map projection makes the spherical geological coordinates fall onto a flat two-dimensional Cartesian coordinate plane. The projection used for the fault system is *New Zealand Map Grid (NZMG)*. The origin of the grid is defined as *latitude 41° 00' 00" South* and *longitude 173°00' 00" East* shown in Figure 2.16. The linear values *False Easting: 2.510*$\times 10^6$ *meters East* and *False Northing: 6.023*$\times 10^6$ *meters North* are added to all x- and y-coordinates, respectively, so that none of the coordinate values in the geographic region which has been mapped are negative.

GNS split the fault lines network on the basis of three attributes, the accuracy, the dominant and sub direction of the movement so that each segment could be described correctly. The accuracy of each mapped fault line is described by *accurate*, *approximate*, *concealed* or *inferred* shown in Figure 2.13. Rattenbury[83] explains that these four values take into account both accuracy of location and certainty of existence:

- ***accurate*** means the position of the fault is known from surface evidence (e.g. outcrop position, air photo expression) to be within 250 m of the marked position.
- ***approximate*** means the position of the fault is not constrained to within 250 m but the fault's existence is not in doubt.
- ***concealed*** means the fault exists but does not have surface expression because it has been buried by a younger deposit. There is no accuracy of location implication here.
- ***inferred*** means the fault's existence is implied by other factors, usually through the spatial relationships of nearby geological map units, but the fault itself has not been seen.

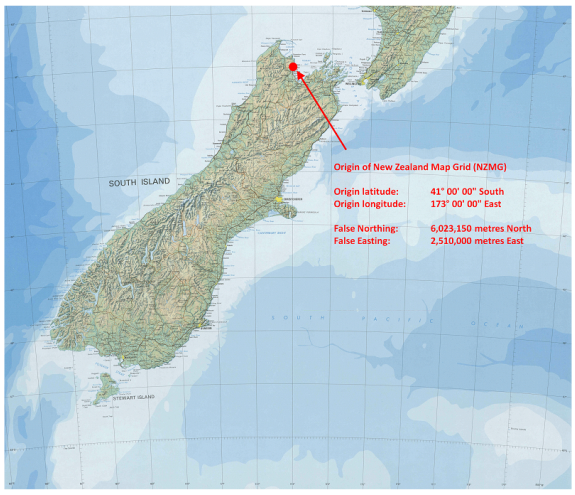


Figure 2.16 ***Origin of New Zealand Map Grid (NZMG).*** *The origin of the NZMG is defined with latitude 41° 00' 00" South and longitude 173°00' 00" East. The linear values False Easting: 2,510,000 meters East and False Northing: 6,023,150 meters North are added to all x- and y-coordinates, respectively, so that none of the coordinate values in the geographic region being mapped are negative.*
(map from LINZ[59])

The dominant and sub direction of the movements can be strike-slip or dip-slip. The splitting of the fault network resulted in more than 27,000 polylines, of a total length greater than 43,000 km. Rattenbury[83] adds that there are regional differences in mapped fault densities - these sometimes reflect the underlying geology but can be due to individual author interpretations (some like to draw more faults than others). Most of these faults are inactive which means that they are unlikely to be involved in producing earthquakes in the near future. Faults that have moved one or more times in the last 10,000 years are commonly considered to be active[88].

2.12 Outline of Research Questions for this Study

In this study, the fractal analysis for the spatial distribution of the fault lines will be carried out by Box-Counting. Using the Box-Counting method, grids of different sizes are superimposed on a fault line map, e.g. one grid with a box size of 4 km, another one with 3 km, 2 km, etc. Zhao, Chen, Zuo and Carranza[112] demonstrate the power law relationship between box sizes and the minimum number of boxes in a log-log graph. This is discussed in more detail in Chapter 3.1. If the hypothesis of a fractal fault system in New Zealand can be supported, the critical state of balance between strain hardening and weakening, and thus at which points ruptures are caused could be determined.

The orientation of fault lines in different areas is investigated by comparing distributions of orientations for different areas. The combination of the *classical Kolmogorov-Smirnov test for two groups (2KS)* and an adjusted version for multiple groups, called the *permutation Kolmogorov-Smirnov test for multiple groups (MKS)*, is used to test whether several geographic areas have statistically significant different distribution of fault line orientation, and where exactly this difference is.

Chapter 3

Fractal Analysis

Fractal geometry is the mathematical equivalent of the paintbrush an artist uses to express the beautiful complexity of nature. Fractals are not new - the world is full of these repetitive patterns. What is new is our appreciation and understanding of these patterns that can be identified at different levels of magnification. The success of bridging the gap between fractal pattern in theory and its application to everyday phenomena is accredited to Benoit Mandelbrot (1924-2010). He was a mathematician and physicist who formulated the term *fractal* in his book *The Fractal Geometry of Nature*[61]. In his words "clouds are not spheres, mountains are not cones, coastlines are not circles, and bark is not smooth, nor does lightning travel in a straight line." These natural shapes cannot be expressed with Euclidean geometry alone as highlighted in Figure 3.1.

Figure 3.2 illustrates the contrast between the smooth world of Euclid and the textured world of fractal nature. In the front we see man-made Euclidean shapes of rectangular and triangular fields, straight roads and cylindrical silos. Behind this Euclidean scenery, we see the Southern Alps of New Zealand with rough mountains rather than triangles with bumps. The rules of fractal geometry are derived from describing natural shapes.

Some natural phenomena are obvious fractals - Mandelbrot[62] explains "when the weather changes and hurricanes hit, nobody believes that the laws of physics have changed". However, he also points to one fractal pattern that is not as obvious and notes that he does not "believe that when the stock market goes into terrible gyrations its rules have changed. It is the same stock market with the same mechanisms and the same people." In general, fractal-based analysis measures how fast length, area, or volume increases, or decreases, with increase, or decrease in scale. Fractals describe the complexity and roughness of a system.

The term fractal derives from the Latin root word *fractum* meaning *break or crack*. In contrast to Euclidean squares and circles, fractals are broken into irregular fragments. Fractal geometry provides a way to measure these fragmented structures. The word fractal geometry consisting of the term *fractal* together with the word *geo* meaning *Earth* and the word *metiri* meaning *to measure* has the interpretation of *to measure the cracks of the Earth*. Exploring fractals is like looking at natural objects from a different perspective.

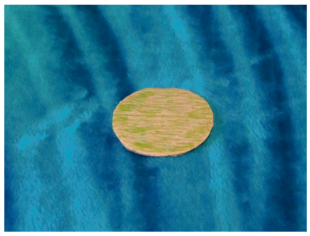
For a long time people could recognize fractal patterns, but their geometric patterns were only explained in the last decades. Points, line segments and spherical shapes can be described by Euclidean geometry because they are easy to construct whereas shapes in nature are more complex. Natural shapes consist of a self-similar pattern on different scales meaning that this pattern is repeated in different sizes and matches the overall shape. Fractal geometry can help to understand the 'how' and the 'why' of these structures. Forms and processes in nature are not the only ones to have a complexity in shape and in time that is not interpretable with traditional Euclidean Geometry. But so are chemical materials, societal structures and music. This ubiquity has resulted in increased development of fractal based ideas in various areas such as in Engineering and in the Arts.

Euclidean Spheres[10]

Natural Clouds[3]

Euclidean Cones[104]

Mountains in New Zealand[6]



Euclidean circle

Coastline of Waiheke and Rangitoto Island in New Zealand[63]

Figure 3.1 ***Illustration of Mandelbrot's explanation between Euclidean and fractal geometry.*** *In his words "clouds are not spheres, mountains are not cones, coastlines are not circles, and bark is not smooth, nor does lightning travel in a straight line." Natural shapes such as clouds, mountains and coastlines cannot be expressed with Euclidean geometry alone. This classical geometry is useful for man-made structures (left), but fractal geometry allows to describe the real world (right).*

Figure 3.2 ***Real life mix between Euclidean and fractal geometry in New Zealand.*** *The mix of Euclidean and Fractal Geometry shown with Euclidean man-made shapes of rectangular and triangular fields, straight roads and cylindrical silos in the Canterbury plains and the fractal shape of the Southern Alp foothills in New Zealand with rough mountains rather than triangles with bumps.*
(figure from Ardmore Pilot[80]*)*

3.1 Fractal Theory

Historically, the term fractal refers to the non-integer dimension. This non-integer dimension stands in contrast to the Euclidean integer dimensions, the whole numbers 1, 2, and 3. While the term *fractal* was only introduced in 1975 by Benoit Mandelbrot, the concept itself has been known for more than one century.

For both Euclidean and fractal dimensions the same concept of calculation can be applied. If we break an Euclidean line segment into N^1 self-similar pieces of the same length, an Euclidean square into N^2 self-similar pieces with the same area and a cube into N^3 self-similar pieces with the same volume, each of the smaller pieces can be magnified by a factor of N to return the original line, area or cube. In other words, the ratio between log number of self-similar pieces and log magnification factor N results in the Euclidean dimension of the object, e.g. for breaking a square into 9 self-similar pieces at magnification factor 3

$$Ed = \frac{\log(N^{Ed})}{\log(N)} = \frac{\log(N^2)}{\log(N)} = \frac{\log(3^2)}{\log(3)} = \frac{\log(9)}{\log(3)} = 2\,,$$

where N is the number of self-similar pieces and Ed is the Euclidean dimension. For the calculation it is not important to which base the log is taken as long as it is the same for the numerator and denominator. In this thesis the natural log is used.

For Euclidean dimensions, we have

$$td = Ed = ed\,,$$

where the Euclidean dimension Ed is equal to the topological dimension td and the embedding dimension ed. The topological dimension td describes how many dimension are needed to cover an object, e.g. we need only two dimensions to cover an area. The embedding dimension ed describes how many dimensions a space needs to place an object in it, e.g. a cube cannot be placed inside a two-dimensional surface but inside a three-dimensional volume.

For fractal objects the same principles apply. One of the best known and most easily constructed fractals is the Cantor Dust illustrated in Figure 3.3. It shows many typical fractal characteristics. If we delete the middle third of a line segment of 1 unit, we get 2^1 line segments, each of the length 3^{-1} units. When we iterate the process we get 2^2 line segments, each of the length 3^{-2} units. So, after each iteration i we have 2^{-i} line segments each of the length 3^{-i} units. All line segments are self-similar but scaled by a factor of $\frac{1}{3}$ between iterations. The more iterations are done, the more gaps occur and the shorter the sum of the line segments' lengths. The number of line segments approaches infinite, but the sum of the length converges to 0. Eventually the line segments decrease to points.

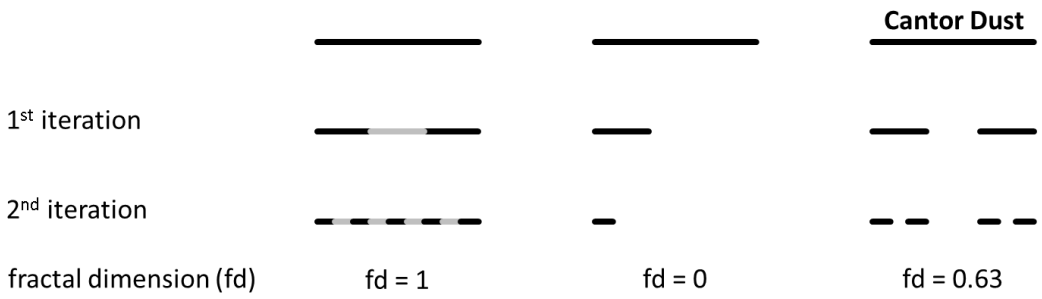


Figure 3.3 ***Concept of calculation for Euclidean and fractal objects.*** *If we break line segments repeatedly into three pieces of the same length, the dimension of the created pattern depends on the number of pieces we keep in each iteration. Keeping one, two or all three parts results in a fractal dimension of fd = 0, fd = 0.63 or fd = 1. Keeping two parts in each iteration is a famous pattern called Cantor Dust.*

But how can we describe this as fractal dimension? When we keep all line segments we would have three line segments scaled by $\frac{1}{3}$ in the first iteration and 9 line segments scaled by $\frac{1}{9}$ in the second iteration. The dimension can be calculated by

$$fd = \frac{\log\left(\frac{N_i}{N_{i+1}}\right)}{\log\left(\frac{s_{i+1}}{s_i}\right)} = \frac{\log\left(\frac{3}{9}\right)}{\log\left(\frac{\frac{1}{9}}{\frac{1}{3}}\right)} = \frac{\log\left(\frac{1}{3}\right)}{\log\left(\frac{1}{3}\right)} = 1\,,$$

where N is the number of fragments and s the scaling factor on the i^{th} iteration. The result is one as expected for a line. When we only keep 1 part in each iteration, we would have one line segment scaled by $\frac{1}{3}$ but also only 1 line segment scaled by $\frac{1}{9}$. The dimension would be calculated by

$$fd = \frac{\log\left(\frac{N_i}{N_{i+1}}\right)}{\log\left(\frac{s_{i+1}}{s_i}\right)} = \frac{\log\left(\frac{1}{1}\right)}{\log\left(\frac{\frac{1}{9}}{\frac{1}{3}}\right)} = \frac{\log\left(\frac{1}{1}\right)}{\log\left(\frac{1}{3}\right)} = 0\,.$$

The result is zero, as the line segment length approaches zero and the whole pattern consist of only this single point. For these two cases the Euclidean dimensions 1 for the line segment and 0 for the point are appropriate. Thus, the dimension when two parts of the line segment are kept must be between 0 and 1. This is the famous Cantor Dust with the fractal dimension calculated by

$$fd = \frac{\log\left(\frac{N_i}{N_{i+1}}\right)}{\log\left(\frac{s_{i+1}}{s_i}\right)} = = \frac{\log\left(\frac{2}{4}\right)}{\log\left(\frac{\frac{1}{9}}{\frac{1}{3}}\right)} = \frac{\log\left(\frac{1}{2}\right)}{\log\left(\frac{1}{3}\right)} = 0.63\,. \tag{3.1}$$

As expected the result is between 0 and 1. In contrast to Euclidean dimension, for fractal dimension we have

$$\begin{aligned} \text{in general: :} \quad & td \leq fd \leq ed \\ \text{for the Cantor Dust:} \quad & 0 \leq 0.63 \leq 1\,. \end{aligned}$$

In addition to the explanation of topological and embedding dimension earlier, we can say that the topological dimension describes how many coordinates we need to describe each part of the pattern. The parts of the Cantor Dust at a high enough number of iteration are two single points. Each of these points does not have any relation within itself and so $td = 0$ for the Cantor Dust. The embedding dimension describes how many coordinates we need to describe the relation between all parts of the pattern. The two points of the Cantor Dust need only one coordinate to describe the distance and so $ed = 1$.

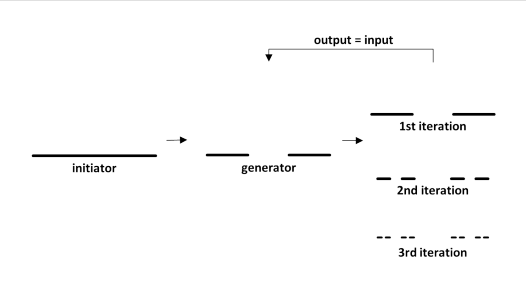
With (3.1) we always get the same fractal dimension for the Cantor Dust, when we compare pattern results of two arbitrary iterations. This characteristic is used to simplify the calculation. We would get the same result if we only took the numbers from one single iteration, e.g. the first iteration

$$fd_S = \frac{\log\left(\frac{N_i}{N_{i+1}}\right)}{\log\left(\frac{s_{i+1}}{s_i}\right)} = \frac{\log\left(\frac{N_i}{1}\right)}{\log\left(\frac{1}{s_i}\right)} = \frac{\log\left(\frac{2}{1}\right)}{\log\left(\frac{1}{\frac{1}{3}}\right)} = \frac{log\,(2)}{log\,(3)} = 0.63\,,$$

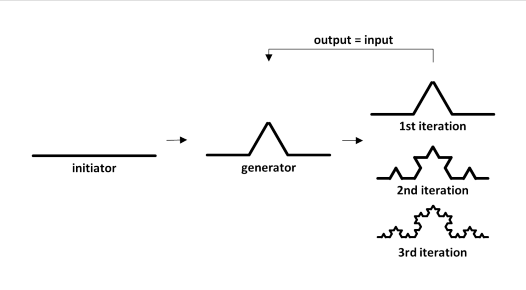
where N is the number of strict self-similar copies of itself, each scaled by the factor s. This definition of fractal dimension fd_S is called the Similarity Dimension. It is a measure of complexity of strictly self-similar fractal

patterns. Later, this will be compared to methods, such as the *Walking-Divider* technique, that estimate the fractal dimension of patterns that are not strict self-similar.

The Cantor Dust is an example for a negative feedback loop meaning in each iteration one part of the pattern is deleted. In Figure 3.4 the initiator and the generator for the Cantor Dust are illustrated. The initiator is a simple line. The generator for the Cantor Dust consists of 2 self-similar shorter line segments scaled by $\frac{1}{3}$. The middle part is erased. This deletion causes the fractal dimension of the Cantor Dust to be smaller than the dimension of the initiator. The fractal dimension of the fractal pattern is 0.63, and the dimension of the initiator is 1.



Negative Feedback Loop for Cantor Dust.



Positive Feedback Loop for Koch Curve.

Figure 3.4 ***Initiator and generator for Cantor Dust and Koch Curve.*** *Depending on the generator for a fractal pattern, the fractal dimension for the same initiator can result for a negative feedback loop in $0 \leq fd \leq 1$, such as fd = 0.63 for the Cantor Dust, and for a positive feedback loop in $1 \leq fd \leq 2$, such as fd = 1.26 for the Koch Curve.*

Natural shapes are not like that. Looking from an airplane, a river appears as only one line. When we magnify this river there may appear smaller rivers at first and later little streams. A geological fault network has the same characteristic of added details during magnifying.

These characteristic seen in natural shapes cannot be described with a negative loop that deletes parts of the original river or fault line. For these natural shape we need a positive loop that adds more and more details to the original shape during magnifying. The Koch Curve is one example for a positive feedback loop, and shows how to get a fractal dimension between 1 and 2, even if we start with the same line segment we had for the Cantor Dust. The Koch curve published in 1906[103] is one of the oldest artificial fractal objects. In contrast to the Cantor Dust, the generator for the Koch Curve consists of 4 self-similar shorter lines, but each line segment again can be magnified by the factor of 3 to produce the original line segment shown in Figure 3.5. The formula

$$\begin{aligned} fd_S = & \quad \frac{log\,(N_i)}{log\,(m_i)} \\ = & \quad \frac{log\,(N_1)}{log\,(m_1)} \quad = \quad \frac{log\,(4)}{log\,(3)} \quad = \quad 1.26\,, \end{aligned}$$

where N is the number of fragments and m is the factor of magnification leads to the fractal dimension 1.26 for the Koch Curve in the first iteration. The magnification factor m is the inverse of the scaling factor s

$$m = \quad \frac{1}{s} \quad .$$

In the next iterations each line segment is replaced by four shorter line segments but in all iterations the log-ratio is the same. The fractal dimension 1.26 is plausible, as the Koch curve is neither a 1-d line segment nor a 2-d surface, and consequently its dimension is

$$\begin{array}{ccccc} td & \leq & fd & \leq & ed \\ 1 & \leq & 1.26 & \leq & 2\,. \end{array} \tag{3.2}$$

For the Koch Curve, the parts are single straight line segments. For that, we need one coordinate to describe the start and end, so *td* = 1. The arrangement of all line segments needs two coordinates and therefore the *ed* = 2.

The relationship between *td*, *fd* and *ed* can be used to check if the calculated result of a fractal dimension is logical. For example, a fault system consists of fault parts whose start and end point can be described with one coordinate. For the complete fault system we need two coordinates. If we calculate the fractal dimension of the whole system we would expect to get a fractal dimension between *tp* = 1 (dimensions of single parts) and *ed* = 2 (dimensions of the whole pattern) as we did for the Koch Curve in equation 3.2. The same is expected for river systems.

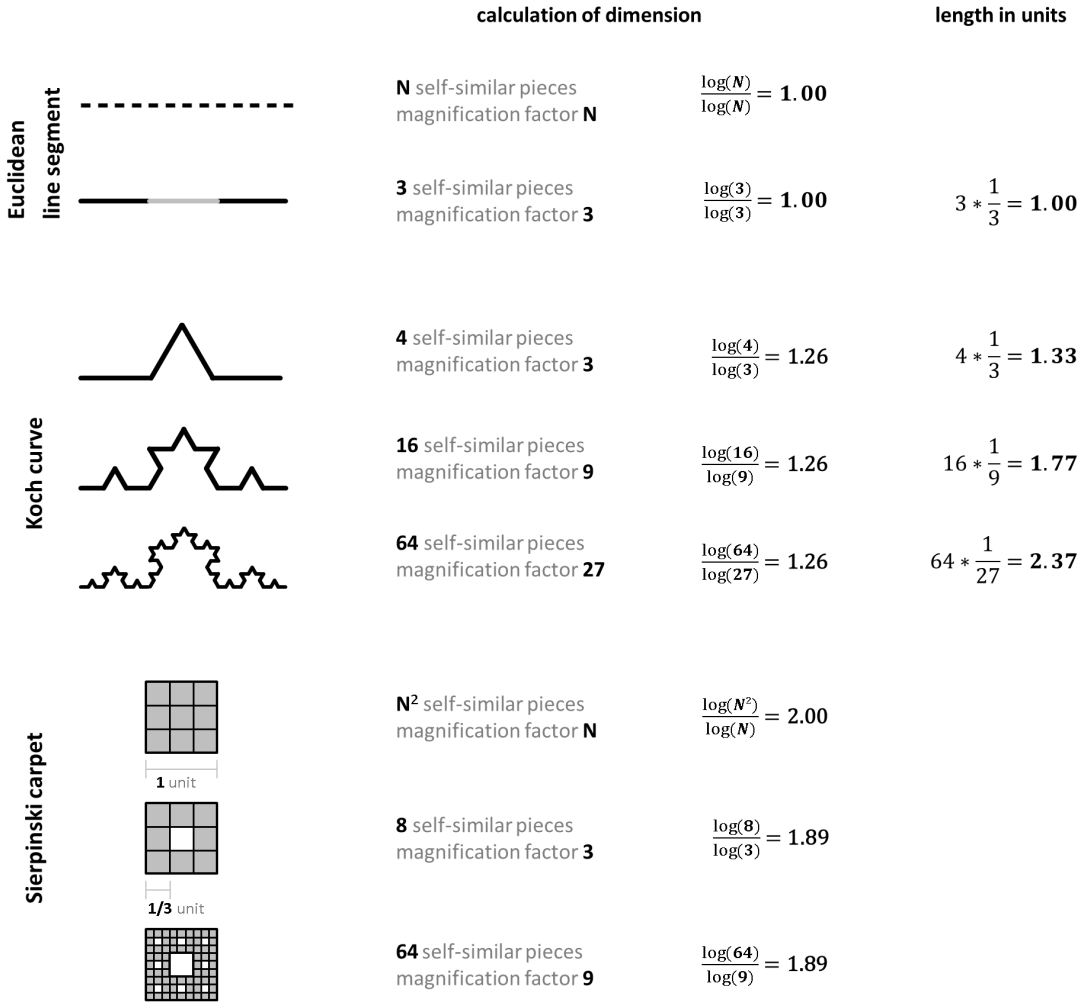


Figure 3.5 ***Koch Curve and Sierpinski Carpet pattern for different iterations.*** *The Koch Curve has a positive feedback loop meaning details are added. The initiator has dimension 1 and the fractal pattern is more complex with the fractal dimension fd = 1.26. The length of the Koch curve increases with the number of iterations, here from 1 unit for the initiator up to 2.37 units for the third iteration. In contrast to the Koch Curve, the Sierpinski Carpet has a negative feedback loop meaning details are deleted. The initiator has dimension 2 and the fractal pattern is less complex with the fractal dimension fd = 1.89.*

A major difference between Euclidean and fractal geometry is the characteristic size scales: whereas Euclidean shapes are normally characterized by size scales such as length, area, volume or radius, fractals do not have such characteristic sizes but are results of iterated formulas and therefore result in different sizes depending on the scale. For the Koch curve in Figure 3.5, the length of the polyline increased from 1.33 units in the first iteration to 2.37

units in the third iteration calculated with

$$l_i \quad = \quad N_i \quad \times \quad s_i \quad ,$$

where l is the length, N is the number of line segments, s is the scaling factor and i is the number of iteration. The length of the Koch curve increases as i increases. As the number of iteration i approaches infinity the length of the Koch curve also approaches infinity. A real life example for the change in size is a coastline of an island. From far away it may look like a circle. From closer we can see more details such as small bays. Bringing these details into account the length of the coastline increases. But there is an important difference between the Koch Curve and the coastline: the Koch Curve is an artificial fractal object with exact self-similarity, whereas the coastline and most of the other objects in real world are only statistically self-similar. There is some variability in self-similarity in natural objects and the pattern may not be an exact match at each scale. For this reason, we cannot use the Similarity dimension that we applied to the Koch Curve.

One of the method we can use is the *Walking-Divider* method. This is also called *yardstick method* method. This method is based on the principle of taking sticks or circles of varying sizes to cover the fractal curve and counting the number of sticks or circles required in each case. Figure 3.6 illustrates the method using circles, where the closest point to the center of the first circle is starting point for the second circle. The numbers of circles for each radius are counted and compared to the radius of the circle. Figure 3.8 (top left) shows the regression line between the log of scaling factor s and the log of the number of pieces for the first three iterations of the Koch Curve. The slope of this line is the negative fractal dimension as

$$fd_{WD}(s,N) \quad = \quad - \frac{\sum_{i=1}^{n}(log(s_i) - \overline{log(s)})(log(N_i) - \overline{log(N)})}{\sum_{i=1}^{n}(log(s_i) - \overline{log(s)})^2}$$

$$= \quad - \frac{\text{cov}[log(s), log(N)]}{\text{var}[log(s)]} \qquad = -(-1.26) \quad = \quad 1.26 \quad .$$

When we fit a regression line between to the log of the scaling factor s and the log of the length l as in Figure 3.8 (bottom left), then the fractal dimension can be calculated as 1 *minus slope*[(60)] as in

$$fd_{WD}(s,l) \quad = \quad 1 \quad - \quad \frac{\sum_{i=1}^{n}(log(s_i) - \overline{log(s)})(log(l_i) - \overline{log(l)})}{\sum_{i=1}^{n}(log(s_i) - \overline{log(s)})^2} \tag{3.3}$$

$$= \quad 1 \quad - \quad \frac{\text{cov}[log(s), log(l)]}{\text{var}[log(s)]} \qquad = 1 \, - (-0.26) \quad = \quad 1.26 \, .$$

Equation (3.3) using the length of the line segments is only suitable for fractals with the topological dimension 1.

A real life example for the change in size is Richardson's (1906) empirical finding of how the measured length of a geographical border changes as the unit of measurement is changed[(86)]. Mandelbrot[(60)] explained Richardson's finding of the change of a coastline's length with the coast of Britain which shows that, the smaller the yardstick, the more detailed the measurement and the longer the coast. Here, I have illustrated it for the South Island of New Zealand in Figure 3.7. For *fd = 1.00* the length of the coast does not change with the length of the yard stick, it remains 2,000 km. In contrast to this, for *fd = 1.10* the coastline lengthens with shorter yard sticks. Decreasing the yardstick from 200 to 50 km increases the coast line's length from 2,000 km to 2,300 km.

The Koch Curve and the coastline consists of only one single polyline. A polyline is a connected series of line segments. Natural patterns, such as those of rivers and faults consist of multiple polylines. These polylines with many start and end points cannot be evaluated using the Walking-Divider technique. For these patterns the *Box-Counting method* is more appropriate. In this chapter I only explain the principle; more details illustrated

	yard stick in units	***m*** magnification factor	***s*** scaling factor	***N*** number pieces	slope (***log(s)*** , ***log(N)***)	***l*** length	slope (***log(s)*** , ***log(l)***)
Koch Curve							
***fd* = 1.26**	$\mathbf{3^{-0}}$ $\mathbf{3^{-1}}$ $\mathbf{3^{-2}}$ $\mathbf{3^{-3}}$	$\mathbf{3^{0}}$ $\mathbf{3^{1}}$ $\mathbf{3^{2}}$ $\mathbf{3^{3}}$	$\mathbf{3^{-0}}$ $\mathbf{3^{-1}}$ $\mathbf{3^{-2}}$ $\mathbf{3^{-3}}$	$\mathbf{4^{0}}$ $\mathbf{4^{1}}$ $\mathbf{4^{2}}$ $\mathbf{4^{3}}$	**-1.26**	**27** **36** **48** **64**	**-0.26**
New Zealand Coast							
fd = 1.00	200 100 50	2^{0} 2^{1} 2^{2}	2^{-0} 2^{-1} 2^{-2}	10 20 40	-1.00	2000 2000 2000	0.00
***fd* = 1.10**	**200** **100** **50**	$\mathbf{2^{0}}$ $\mathbf{2^{1}}$ $\mathbf{2^{2}}$	$\mathbf{2^{-0}}$ $\mathbf{2^{-1}}$ $\mathbf{2^{-2}}$	**10** **22** **46**	**-1.10**	**2000** **2200** **2300**	**-0.10**
fd = 2.00	200 100 50	2^{0} 2^{1} 2^{2}	2^{-0} 2^{-1} 2^{-2}	10 40 160	-2.00	2000 4000 8000	-1.00
	box edge length in units						
Sierpinski Carpet							
***fd* = 1.89**	$\mathbf{3^{-0}}$ $\mathbf{3^{-1}}$ $\mathbf{3^{-2}}$ $\mathbf{3^{-3}}$	$\mathbf{3^{0}}$ $\mathbf{3^{1}}$ $\mathbf{3^{2}}$ $\mathbf{3^{3}}$	$\mathbf{3^{-0}}$ $\mathbf{3^{-1}}$ $\mathbf{3^{-2}}$ $\mathbf{3^{-3}}$	$\mathbf{8^{0}}$ $\mathbf{8^{1}}$ $\mathbf{8^{2}}$ $\mathbf{8^{3}}$	**-1.89**		

Table 3.1 ***Walking-Divider and Box-Counting fractal dimension for Koch Curve, Coastline of New Zealand and Sierpinski Carpet.***
Koch Curve: *The slope of a line fitted to the log of the scaling factor s and the log of N is the fractal dimension with sign reversed, here it is fd = -(-1.26) = 1.26 for the Koch Curve. With decreasing scaling factor s, the length l of the Koch Curve increases. The fractal dimension is the slope of the line fitted to log(s) and log(l) after subtracting from 1, for the Koch Curve it is fd = 1-(-0.26) = 1.26.*
Coastline of New Zealand: *The number of self-similar pieces needed to cover the coast of the South Island of New Zealand increases from 10 to 22 and 46 using yardsticks of the size 200, 100 and 50 km, respectively. The relation is not proportional and results in different total lengths of the coast line. A fractal dimension of 2 would even have a stronger effect on the coast length, but a fractal dimension of 1 would not effect the total length.*
Note, the Koch Curve's N increases by the factor of 4 in each iteration, whereas N for the Coastline does not increase exactly by a factor. Most fractals in nature are similar to the Coastline meaning they are not exact self-similar.
Box-Counting fractal dimension for Sierpinski Carpet: *We can calculate the fractal dimension fd as slope of a line fitted to log(s) and log(N) with sign reversed. Even simpler, we can use the ratio between log of occupied boxes and log magnification factor such as* $\frac{log(8)}{log(3)}$ *= 1.89.*

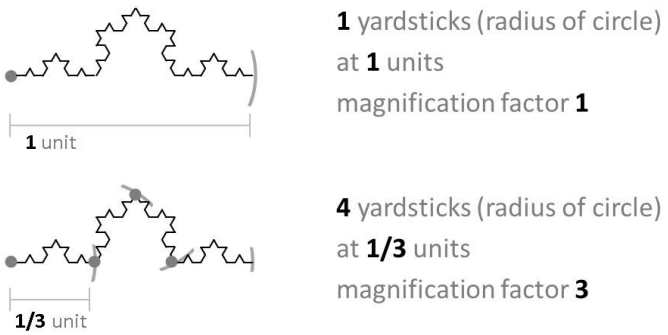


Figure 3.6 ***Walking-Divider technique at Koch Curve.*** *The Walking-Divider technique uses yardsticks or circles of different sizes. The closest intersection of the circle with the Koch Curve to the center of the circle itself is starting point for the next circle. The number of circles are counted and compared to the radius of the circle. We need four circles at a radius of $\frac{1}{3}$ unit for the Koch Curve in the first iteration.*

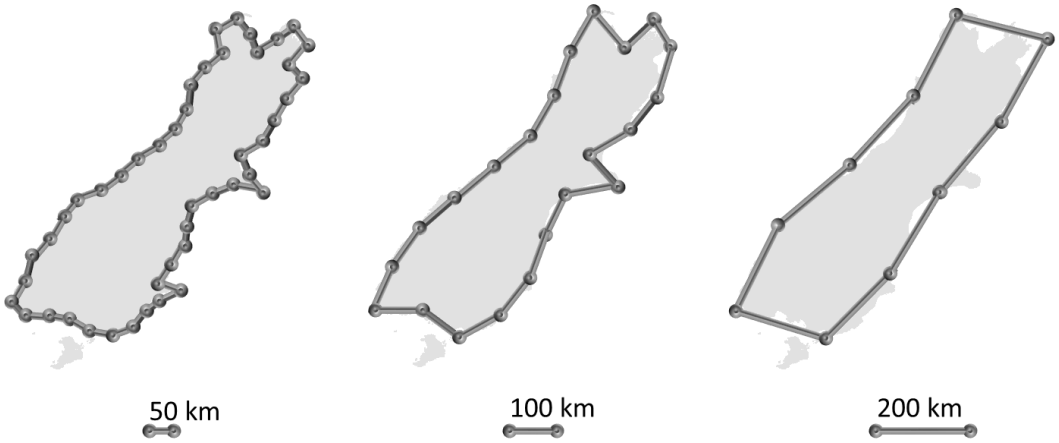


Figure 3.7 ***Increasing length of coastline in New Zealand with shorter yardsticks.*** *The measured length of the coast of the South Island of New Zealand depends on the unit of the yardstick. The smaller the unit the more precise the measure resulting in an increasing length. The measurement with yard sticks of the length 50km, 100km and 200km results in a estimated fractal dimension of fd = 1.10.*

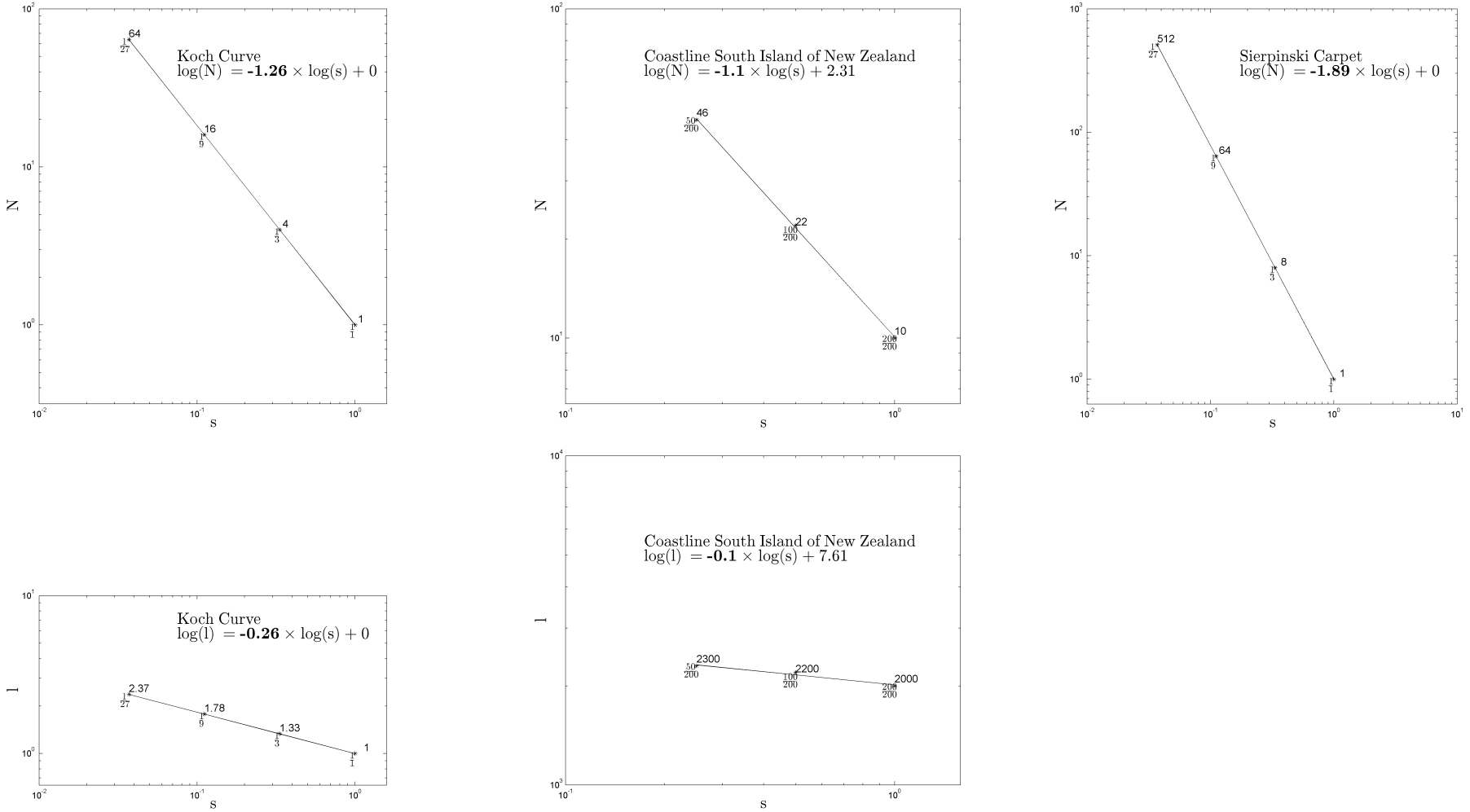


Figure 3.8 ***Walking-Divider and Box-Counting fractal dimension for Koch Curve, Coastline of New Zealand and Sierpinski Carpet.***
top: *For all three pattern the fractal dimension can be estimated by the slope of the regression line fitted to the log of the scaling factor s and the log of the number of yardsticks or boxes N with sign reversed, e.g. fd = -(-1.26) = 1.26 for the Koch Curve. The upper integers are the number of yardsticks N, and the lower numbers are scaling factors s. Here, the natural log is used for all calculations.*
bottom: *For the Koch Curve and Coastline the fractal dimension can also be estimated by the slope of the regression line fitted to log of the scaling factor s and log of curve or coast length l, when we set the fractal dimension fd = 1 - slope, e.g. fd = 1-(-0.26) = 1.26 for the Koch Curve. The upper numbers are the lengths l and the lower numbers are the scaling factors s.*

with an application to an artificial fractal object are provided in Chapter 4. Figure 3.5 illustrates the process of Box-Counting using the Sierpinski carpet. The process involves splitting the carpet into boxes and counting the number of boxes used to describe the pattern. This process is iteratively repeated with boxes of different sizes so that the rate of change in complexity with scale can be found. The Box-Counting dimension fd_B is given by

$$fd_B = = \lim_{i \to \infty} \frac{log(N_i)}{log(m_i)} \quad ,$$

where N is the number of covered boxes and m the magnification factor. The magnification factor m determines the box edge length, here for the Sierpinski carpet in the first iteration

$$\begin{aligned} e_i &= = \frac{k}{m_i} \\ e_1 &= = \frac{1}{3} \quad unit \quad , \end{aligned}$$

where e is the box edge length and k is the edge length of the total carpet (1 unit). For the Sierpinski carpet we calculate for the first iteration

$$fd_B = = \frac{log(N_i)}{log(m_i)} = = \frac{log(8)}{log(3)} = 1.89 \quad .$$

In the case where there is no change in the dimension across the scale, the pattern is called fractal. We usually focus on the fractal dimension of the limit where the magnification goes to infinity when the box edge length goes to 0. Due to the fact that the ratio for different scales for an artificial exact fractal, such as the Sierpinski carpet, is the same we also can use the slope of the regression line fitted to log of box size in units and log of covered boxes as a measure for the estimated fractal dimension. This slope is the negative fractal dimension and is illustrated in Figure 3.8 and Table 3.1.

On first sight Box-Counting looks similar to zooming in and out like with a camera as both methods can be used for describing the complexity of a pattern on different scales. However, zooming shows the pattern from closer or further away and keeps the size of area described by one box whereas Box-Counting changes the size of area described by one box.

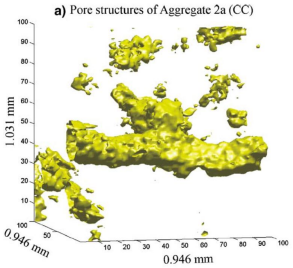
3.2 Fractal Examples

Fractal analysis has been constructively supportive in a great range of research areas such as biophysics, material science, engineering and medicine. A review is provided here.

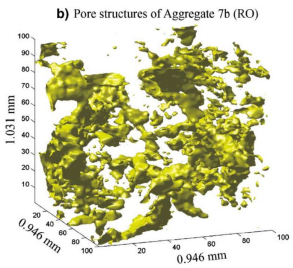
3.2.1 Biophysical and Biochemical Fractals

In the field of soil physics, fractals have been effective in characterizing heterogeneity and complexity of the soil characteristics. Kravchenko, Wang, Smucker & Rivers (2011)[55] investigated how the tillage and land use differences affect pore heterogeneity of soil aggregates using fractal analysis based on occupied boxes. They covered three-dimensional images that were segmented into pores and solids with boxes of varying sizes that represent different scales. They compared the number of boxes of different scales containing pores and their result suggests that long-term differences in land utilization and management practices may lead to substantial differences in the pore-distribution and structure of soil aggregates.

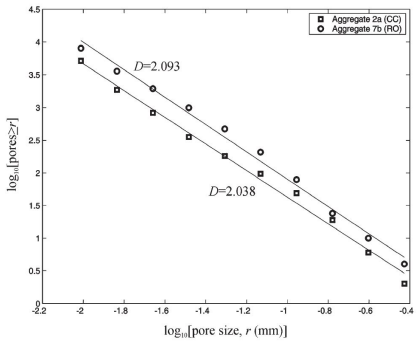
Instead of focusing on the pores, Gibson, Lin & Bruns (2006)[36] compared internal heterogeneity of individual aggregates from soils of continuous corn and soils of crop rotation treatment. They placed numerous boxes of different sizes and calculated the density as total aggregate mass divided by the solid volume. Their results indicated that the crop rotation tended to increase soil porosity, soil structural hierarchy, and aggregate stability illustrated in Figure 3.9. In contrast to soils of agricultural areas in that study, Zhou, Peng, Peth & Xiao(2012)[113] compared soil of severely eroded bare land and vegetation restoration land. They expected that vegetation restoration improved soil microstructure and therefore enhanced soil stability and reduces soil erosion. They used fractal theory and found that the fractal dimensions were statistically significantly higher in the restoration land than in eroded bare land, indicating improved pore system after vegetation restoration.



pores of soils of continuous corn (CC)



pores of soils of crop rotation treatment (RO)



The soil structure of rotation treatment (RO) is more complex than of continuous corn (CC).

Figure 3.9 ***Pores of soils dependent of the corn treatment.*** *The internal heterogeneity of different soils was compared by placing three dimensional boxes of different sizes and calculating the density. The fractal dimension for rotation treatment fd = 2.093 is higher than for continuous corn treatment fd = 2.038. The result is that corn rotation treatment tended to increase soil porosity, soil structural hierarchy, and aggregate stability.*
(from Gibson, Lin & Bruns[36])

3.2.2 Material Science

Some examples from biology and engineering focusing on material characteristics are as follows. The pore structures of eggshells plays a crucial role in protecting the contents of the egg from the microbial and physical environment and in controlling the exchange of water and gases. Zhang & Wang (2012)[111] determined the Box-Counting fractal dimension of pore distribution. They could easily distinguish the blunt end of the eggshell from the sharp

end by differences in the pore-distribution. This difference in the pore-distribution allows saturated salt solution to diffuse differently in these areas into the egg.

Fractal analysis has also been useful for analyzing artificial materials. Many artificial materials are chemical compounds that consist of repeating structural units, called polymers. These polymers are used for fabricating electronic devices from low-cost materials which in turn allow the production of low priced electronics. Kleiner, Suchaneck, Adolphi, Ponomareva & Gerlach (2012)[52] investigated ceramic film deposition onto these compounds, a process to improve their mechanical resistance. They found different fractal dimension for the ceramic film depending on its composition and deposition condition. Using enhanced technology for depositing the ceramic film results in a higher fractal dimension that represents an increasing surface roughness.

This study agrees with Kobayashi, Maruyama, Tsurekawa, & Watanabe (2012)[53] who investigated the effect of grain boundary microstructure on the fracture resistance of a chemical element. They analyzed the fractal dimension of the grain boundary network with Box-Counting. They found that the fracture toughness and the fracture strength of brittle polycrystalline materials increase with increasing fractal dimension of fracture surface and propose that the fractal dimension can be used effectively to predict and control fracture in polycrystalline materials.

Similar to these solids, fractals are also applied to aggregates after phase transition from fluid to solid. Crivoi & Duan (2012)[24] attempted to predict the formation of nanoparticles from fluids induced by evaporation. They dried droplets of different nanofluids at room temperature and observed that the remaining fractal-like aggregates agree with their simulation results.

3.2.3 Medical Fractals

Fractal analysis is also widely used in medicine. The fractal dimension of individual unwell people's organs is estimated for diagnosis. Human bodies have many fractal organs used to maximize both surface area and flow. Lungs have a fractal pattern that allows to exchange much more oxygen with the blood than a non-fractal lung could do. The amount of gas that can be exchanged in the lungs is proportional to their total surface area. Fractal geometry allows a very large surface area to be extremely compact. With a fractal model kidneys can filter more waste and the neurons can connect with more neurons. Figure 3.10 compares a fractal model of a neuron with a real cell dendritic branching.

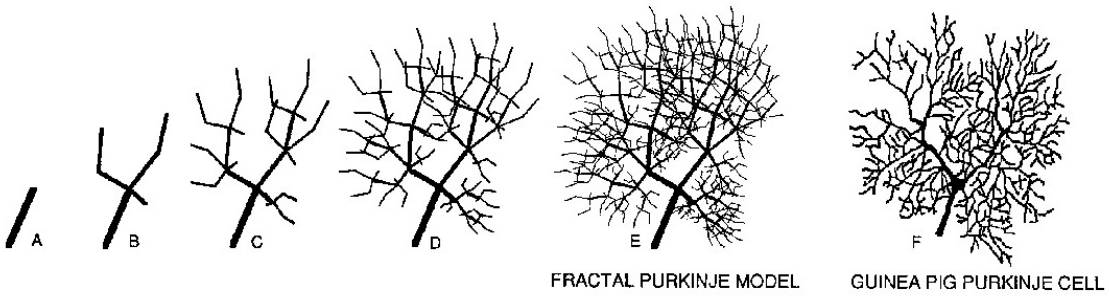


Figure 3.10 ***Demonstration of the practicability of fractal models to reduce and preserve complex branching.*** ***A-D*** *show an initial trunk and three positive loop iterations of fractal growth using the generator* ***B****. Comparison of B and C illustrates the replacement of each branch by the fractally proportioned generator* ***B****.* ***E*** *shows the neuron model after the* 4^{th} *iteration, which is to be compared with the real cell dendritic branching in* ***F****.*
(figure from Pellionisz[79]*)*

Yan & Guo (2012)[108] illustrated the value of fractals in health application by studying the computational fractal dimension of human colonic pressure activity using the Box-Counting method. Their results show that pressure activity might be fractal after it is integrated over time and suggest that fractal estimation might provide a new method to better understand the nonlinear dynamics of human gastrointestinal pressure recordings.

Kalda[47] proposed a model of the human blood-vessel system that is consistent with the processes governing the growth of the blood-vessels. This fractal model can be used for analysing the transport of passive component by blood.

Comparing the fractal dimension of a body part of one individual to the population distribution can support diagnosis of problems in lungs, blood vessels, nervous diseases etc. In contrast to this physiological applications, Hong & Huidong (2012)[46] focused on the anatomy. They calculated the fractal dimension from the texture feature extraction of x-ray chest images for image retrieval. They recommend their fractal algorithm to be applied not only to image retrieval systems, but also to image auto-classification systems to assist doctors managing medical images.

3.2.4 Societal Fractals

Fractals are also helpful for examining infrastructures and societal principles. Zhang & Li (2012)[110] investigated the fractality and self-similarity in the structure of road networks for better understanding of the complexity and dynamics of the road system. Instead of a geometric representation of the roads they used a structural representation that is highly abstract and only concerns the nodes and links of the roads. By covering the network structure with boxes of different sizes they found that road networks have fractal structure. This knowledge provides an empirical guide for urban design and transportation planning.

An example of a crucial organizational principle is the protection of communities by providing information to police officials about the intensity of crimes' pattern distribution. Sridhar & Balasubramaniam (2012)[93] estimated the fractal dimension of the spatial distribution of crimes using an extended triangularisation and Box-Counting algorithm and could show that the fractal dimension increases or decreases as perimeter of polygon changes. They suggest that their model can be used for any geo-referenced point data such as cancer incidence data and hypertension data.

For the police, fractal analysis not only helps to manage crime but also to support forensic analysis. Konopinski, Hudziak, Morgan, Bull & Kenyon (2012)[54] found that sand grain surfaces exhibit a statistically self-similar fractal nature that remains the same across scales. They believe that this quantitative measure could be employed in the discrimination of grains based on their place of origin. In authors' opinion this can help in criminal investigation and consequent court proceedings.

3.2.5 Biological Fractals

Even if we are not thinking of organic structures as geometrical shapes, they do have a characteristic geometry. In the 16th century Leonardo da Vinci noted that "All the branches of a tree at every stage of its height when put together are equal in thickness to the trunk"[81]. If a mother branch produces two daughter branches, the sum of the cross-subsections of the daughters equals the area of the cross-subsection of the mother, mathematically expressed

$$\begin{aligned} \pi \times r_m^2 &= \sum_{i=1}^{d} \pi \times r_i^2 \\ r_m^2 &= \sum_{i=1}^{d} r_i^2 \end{aligned} \quad ,$$

where r_m is the radius of the mother and r_i are the radii of the d daughter branches (Figure 3.11). Eloy explains that depending on the geometry of the specific species of tree the exponent in the equation that describes Leonardo's hypothesis is not always equal to 2 but rather varies between 1.8 and 2.3[28]. They suggest that this pattern is the reason why trees do not splinter.

Similarly, a fern is made up of progressively smaller limbs. These smaller limbs resemble the whole fern leaf. Since each smaller subsection of a fractal resembles the whole pattern, they are called self-similar. The older the

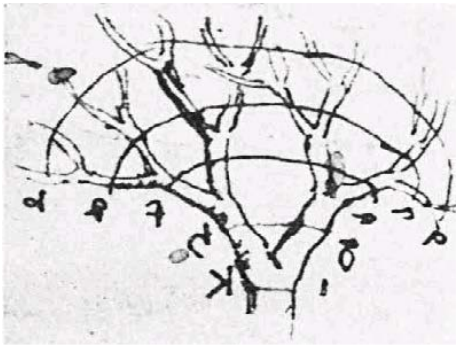
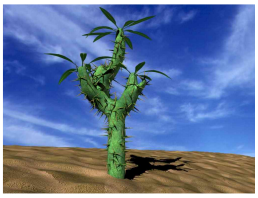


Figure 3.11 ***Leonardo da Vinci sketched the branching pattern of trees.*** *He suggested that the total cross section of daughter branches along each of the arcs would equal the cross section of the trunk.*
(figure from Richter 1939, plate 27 in Aratsu[16])

fern[13] (highly complex)

tree[12] (highly complex)



cactus[7] (low complex)

Figure 3.12 ***Increase of amount of photosynthesis with increasing Surface Area of Plants.*** *Increased surface area due to fractal structure extend the amount of photosynthesis. Plants in the desert need to conserve water and their surface area is therefore less complex.*

Linear spiral[8]

Linear spiral of garden hose[2]

Logarithmic spiral[8]

Logarithmic spiral of Koru Fern[5]

Figure 3.13 ***Contrast of linear and logarithmic spirals.*** *Compared to linear spirals, logarithmic spirals of organic structures show self-similarity on different scales. The Koru Fern is one of the plants that represent with its logarithmic spiral self-similarity that is one characteristic of fractals.*

fern and the bigger it is the more complex is this self-similar pattern.

What is the evolutionary advantage of these fractal structured plants? The more complex the fractal structure the bigger the surface area shown in Figure 3.12. This increased surface area enables the plant to convert more light energy, normally from the sun, into chemical energy for the organisms' activities. In contrast to the tree and the fern, a cactus in the desert needs to conserve water. The surface area of a cactus is smaller and therefore less complex than that of a tree or a fern[(25)]. A further benefit of a fractal structured plant may have been the compact plan of growth - it is a self-repeating plan on different scales. Plants with such a compact plan may have had an evolutionary advantage compared to plants whose plan of growth had to describe each detail separately.

Another characteristic of fractals can be seen at the fern's spiral. When you roll a garden hose you get a linear spiral shown in Figure 3.13. The hose has the same distance between each revolution. But plants tend to produce logarithmic spirals, which decrease the distance between each revolution[(25)]. The Koru Fern for example, has a passive stem and the growing tip is a self-replicating logarithmic spiral.

Besides organic structures many natural processes have fractal patterns. The pattern of natural processes are often characterized by percolation changing systems[(32)]. In these systems, opposing forces of growth and inhibition strive to establish equilibrium. When theses forces struggle for balance, often a fractal structure emerges, in which filled and empty spaces have a distinct self-similar pattern (see also self-organized criticality in Section 2.8. The organic process by which the fruit production of trees increases and decreases can be explained by such opposing forces. In a French orchard in the 15th century[(105)], a group of French monks experimented with tree formations in order to harvest the most fruit. They started with straight rows to allow maximum number of trees per areas. However, this tree formation allowed fruit eating insects easily to travel from one tree to the next. Fixing this insect problem by planting the trees further apart from each other resulted in fewer trees and therefore smaller harvest. The monks planted trees in clusters, decreased and increased the gaps between them depending on whether the trees were attacked or not. The most successful pattern they observed over the decades was a complex pattern of clusters and gaps with no apparent uniformity in structure shown in Figure 3.14. Today we explain the monks' orchard pattern with fractal geometry. Most present orchards do not have this fractal pattern. But in the time of higher awareness of health issues and the return to organic food, this fractal formation of trees may attract fruit producers' interest again as an alternative to the use of pesticides in organic fruit production.

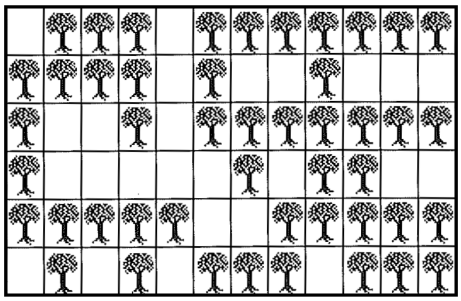


Figure 3.14 ***Monks' fractal orchard.*** *Monks in the fifteenth century experimented with different pattern of planting trees to fix the insect problem and increase the harvest. This intricate pattern of clusters and gaps with no apparent uniformity is fractal and was used to get the most fruit from trees.*
(figure from Wahl[(105)])

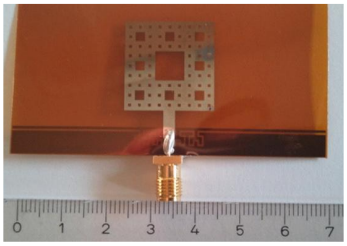
3.2.6 Engineering Fractals

A light bulb's brightness is directly related to the compactness of exposed filament. A filament transposes electrical power into light and heat energy. With dense package of filaments the spirals heat each other producing a higher temperature and higher brightness. This packing pattern is traditionally achieved by coiling the filament twice into spiral loops. There are small and big spirals. Each size represents one scale. The result is a fractal pattern on two

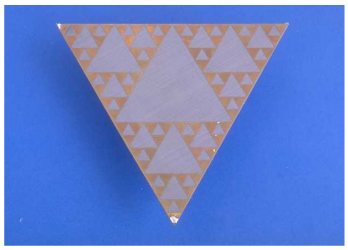
scales illustrated in Figure 3.15. Therefore fractal pattern can save power as for the improvement of the brightness no additional power is required.

Figure 3.15 ***Light Bulb's fractal spiral loops.*** *Fractal wounding of the filament increases the length of the filament and improves the illumination. Here, we have to sizes of spirals that represent a fractal pattern at two scales. (figures from planet schule[11])*

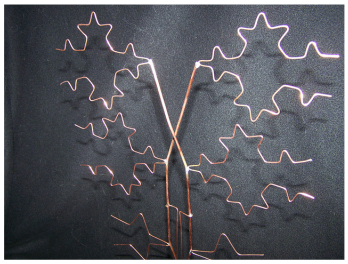
Another everyday product is fractal antennas. Fractal antennas can overcome one common design problem where the antenna is sensitive to only a narrow range of frequencies. Fractal antennas with only a small number of iterations of a fractal process can be sensitive at several frequencies. The reason is the self-similarity of the fractal pattern that provides for each wavelength a good effectiveness. This is very important as different technologies use different wavelength. With traditional antennas there was often a need for more than one antenna but a single fractal antenna can be used for different technologies such as WLAN and GPS. They are suitable for small devices such as cell phones as their space filling pattern allows a small size. Figure 3.16 illustrates that fractal antennas patterns often based on mathematical designs such as the Sierpinski Carpet, the Sierpinski Triangle and the Koch curve. The Sierpinski Triangle is based on the same principle as the Sierpinski Carpet.



antenna in a cell phone based on the Sierpinski carpet[82]



antenna based on the Sierpinski triangle[1]



fractal TV antenna based on the Koch Curve[14]

Figure 3.16 ***Fractal antennas based on Sierpinski Carpet, Sierpinski Triangle and Koch Curve.*** *Fractal antennas offer similar reception over many different wavelengths due to similar structure at different scales. Because of their compact construction, they are popular for small devices.*

3.2.7 Environmental Fractals

Scientists study epidemics by analyzing the factors that control the growth and containment of a disease. For example, diseases are more likely to spread in urban areas with a high density of population than in a rural areas with many kilometers of distance between neighbours. Knowing the infection pattern helps to decide who to protect by quarantine or vaccination. Kendal[49] found evidence for a fractal stochastic process underlying measles epidemics in Britain. He examined the variability in measles incidence during the pre-vaccination period of 1944 to 1966. He found a power-law scaling of incidence time-series and the distribution histogram showed a fractal geometric distribution. He concluded that the development and resolution of measles epidemics are influenced by fractal stochastic processes.

3.2.8 Fractals in Arts

Fractals are not only in Nature and in Science - they also appear in the arts. Taylor, Micolich & Jonas analyzed Pollock's drip paintings and showed that he refined his technique over the years [95]. Through the years of his work the fractal dimension of his painting increased continuously as illustrated in Figure 3.17. The distinct change in the fractal dimension with time could be used as a quantitative, objective way both to validate and date Pollock's drip paintings.

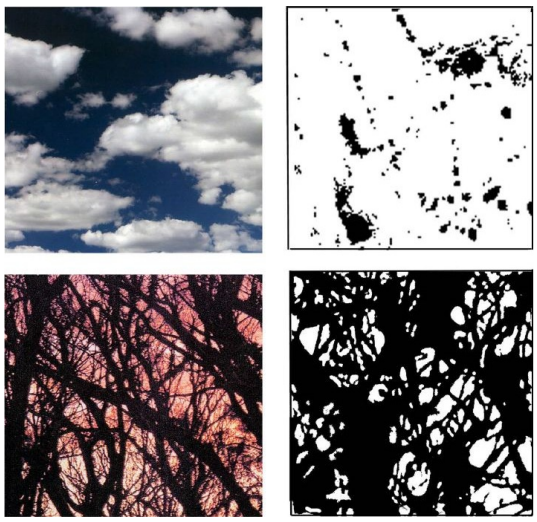


Figure 3.17 ***Fractal dimension of natural scenery compared to Pollock's paintings.***
top: *Clouds (left) and Pollock's painting "Untitled" (1945) (right) have a low fd = 1.3 and fd = 1.10, respectively).*
bottom: *A forest (left) and Pollock', painting "Untitled (1950)" (right) have a high fd (both have fd = 1.89).*
(figure from Taylor [96])

Repeated pattern of different scales appear also in Architecture. One example is Kandariya Mahadeva Temple in India that is part of the Indian UNESCO World Heritage *Khajuraho Group of Monuments* [99]. The temple is a preserved fractal building from the mid-11th century. The fractal pattern consists of the main spire with its 84 adorning fractal replicas shown in Figure 3.18.

Kandariya Mahadeva Temple in India [4] *Pont du Gard in France* [9]

Figure 3.18 ***Fractal buildings in Asia and Europe.***
left: *The temple has a fractal pattern consisting of the main spire and its 84 adorning fractal replicas.*
right: *The Roman aqueduct built in the 1st century shows a fractal pattern on three scales.*

Even earlier, the Roman aqueduct Pont du Gard built in France in the 1st century shows a fractal pattern on three scales displayed in Figure 3.18. Eventually this bridge was registered as UNESCO World Heritage [100], whether it is because of its fractal architecture, hydraulic engineering system or both.

Not only in Asia and in Europe but also in Africa one finds fractal architecture. In Africa it is not a single building but the pattern of villages that is fractal. One example is the village Ba-ila in southern Zambia that

displays social status in the architecture shown in Figure 3.19[27]. Both geometric elements of this structure, the overall ring shape and the status gradient, are repeated on each scale of the settlement. In the back area of each family ring there are the family living quarters. In the front is a gate for letting livestock in and out. The unclean front yard with the animals is associated with a lower status than the clean back area where the people live. This scaling of social status is reflected by the scaling in the architecture of each family ring: no building in the front that represent low status, smaller building for storage in the middle and larger houses in the back representing high status.

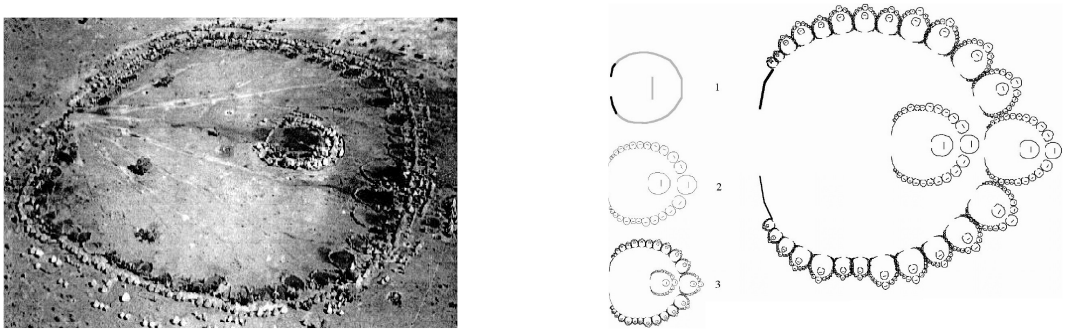


Figure 3.19 ***Fractal settlement in Africa.*** *First iteration of the fractal pattern of the village is similar to single house, second is similar to family ring, third to village as whole.*
left: *Real Ba-ila village in Zambia*
right: *The fractal generation of the settlement*
(figures from American Geographic Institute and Eglash[27])

3.2.9 Geological Fractals

Scale-invariance is one characteristic of fractal patterns that is commonly known from geological objects. In Figure 3.20 you would not be able to estimate the size of the rocks without the pencil. This is the reason why geologists include a pencil or other measurement on a photo of a rock.

Pillow basalts in Oman with pencil for scale[65]

Upright Syncline in Limestone with a pick for scale[67]

Figure 3.20 ***Scale invariant rocks.*** *Rocks have the same pattern on different scales. This is the reason why Geologists take photos including measurement such as a pencil or a hammer as reference.*

Figure 3.21 illustrates this difficulty of size estimation with the fractal pattern of a river system. Small streams merge into larger streams, large streams merge into rivers. Faults are connected in a similar way. Fractal fault systems are already mentioned in Section 2.7.

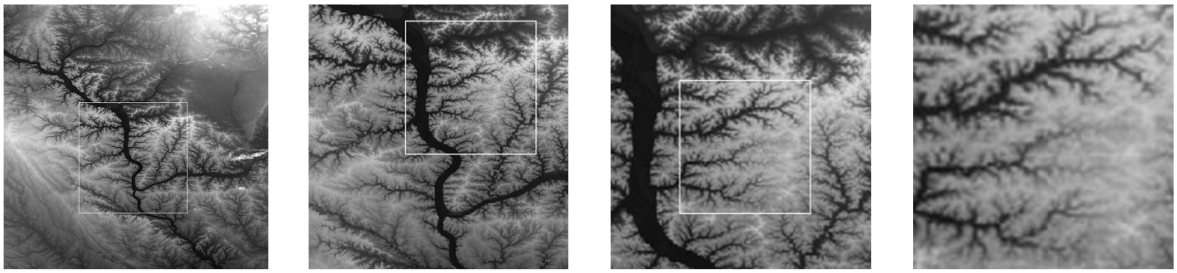


Figure 3.21 ***Fractal river system in the USA*[56].** *The river system "Driftless Area" in southwestern Wisconsin shows the same pattern on different scales. Each white framed square shows the zoomed-in area for the next image.*

3.3 Summary

In the beginning of the chapter the difference between Euclidean and fractal geometry was explained. One difference is the integer dimension for Euclidean objects, whereas fractals have a non-integer dimension. Another major difference is the characteristic size scales. Euclidean object can normally be described by size of length, area or volume. In contrast, the length, area and volume for fractal objects changes with scales. If the size decreases with larger scale, it is a negative loop such as for the Cantor Dust. If the size increases with larger scale, it is a positive loop such as for the Koch Curve.

There are different definitions for fractal dimensions. The Similarity dimension can be used for exact self-similar fractals such as the Koch Curve. Most fractal objects in real world such as coastline and a fault system do not have an exact match at each scale . If these not strict self-similar objects have topological dimension $td = 1$ and consist of only one polyline, we can apply the Walking-Divider technique using circles or yardsticks of different sizes.

A fault line system consists of multiple polylines rather than only one such as the Koch Curve and Coastline, and it is not an exact self-similar fractal pattern. For the fault lines Box-Counting is appropriate. The concept of Box-Counting was illustrated with the Sierpinski Carpet.

In this chapter I have outlined the most common examples of fractal applications in science, medicine and other areas. Finally I explained the scale invariance with geological fractal such as rocks and show a positive loop with a fractal river system.

I now focus on the theory of Box-Counting in Chapter 4 and the case study of New Zealand in Chapter 5.

Chapter 4

Box-Counting Methodology and Procedures

In this chapter the methodology and procedures of Box-Counting are reviewed. It starts with a short example of the Box-Counting fractal dimension estimation, before continuing with a detailed description of all steps involved in Box-Counting. The steps are illustrated using the Koch Curve and its known fractal dimension to provide a reference point so that the importance of crucial user-made choices in the application of Box-Counting becomes evident.

The origin of Box-Counting can be traced back to the Sierpinski carpet introduced in 1916 [90] introduced by Waclaw Sierpinski, a Polish mathematician. The construction of his famous fractal pattern, the Sierpinski carpet, starts with a square, which is the initiator (compare initiator of Cantor Dust and Koch Curve in Figure 3.4). In the first iteration, the square is divided into 9 sub-squares of equal size. The central sub-square is removed as illustrated in Figure 3.5. In the following iterations the same procedure is recursively applied to the remaining 8 sub-squares. The pattern of the remaining 8 squares is the generator. For each sub-square size the number of occupied boxes is counted. The relation between counts and box sizes is the basis for estimating the fractal dimension.

The fault lines of the South Island are not in box form. However, the lines can be covered with a grid and the boxes containing a fault line can be counted as occupied boxes. For estimating the fractal dimension the same approach is used as for the Sierpinski carpet

$$fd_{LR(i)} \quad = \quad = \lim_{i \to \infty} \frac{log(N_i)}{log(m_i)} \ ,$$

where $fd_{LR(i)}$ is the Box-Counting fractal dimension, N_i is the number of occupied boxes and m_i is the magnification factor of the ith iteration of the scaling process. The formula states that we are usually interested in the fractal dimension of the limit where the magnification tends to infinity and the box edge tends to 0.

However, for real systems, such as the fault system on the South Island which covers an area of hundreds of square kilometres the available computer capacity does not allow numerical calculation of very small box edges. It is sufficient to increase the magnification factor until either a convergence of the fractal dimension is reached or until one can say that there is no convergence.

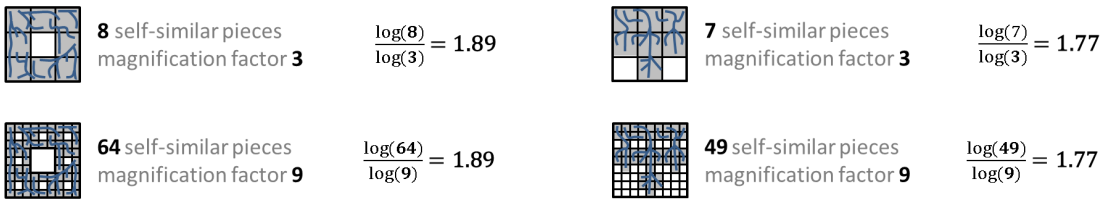
From the geological perspective, we are not interested in small box sizes, such as 1 m^2 or smaller. The reason for this is that not all fault lines are accurately mapped; some are approximate, concealed or inferred. Furthermore, the concept of band-limited fractal structures does not use the concept of the limit. Band-limited fractal structures have different fractal dimensions for different scale ranges, e.g. one fractal dimension for boxes smaller than 100 m^2 and another one for bigger boxes.

In Figure 4.1 (left) the estimation of the fractal dimension for the artificial fault network for two scales with

relatively large box sizes, 8 boxes scaled down by $\frac{1}{3}$ and 64 boxes scaled down by $\frac{1}{9}$ is calculated by

$$\text{for the first iteration:} \quad fd_{LR(i)} = \frac{log(N_i)}{log(m_i)} = \frac{log(8)}{log(3)} = 1.89$$
$$\text{and for the second iteration :} \quad d_{LR(i)} = \frac{log(N_i)}{log(m_i)} = \frac{log(64)}{log(9)} = 1.89\ .$$

For these artificial networks in Figure 4.1 we get exactly the same result at each scaling. In contrast, fractals in nature are never exactly self-similar. There will always be some variation due to the stochasticity in natural systems. Another source of variation is from the actual process of estimating the fractal dimensions.



The fault lines cover 8 out of 9 and 64 out of 81 boxes resulting in a estimated fractal dimension of 1.89.

The fault lines cover 7 out of 9 and 49 out of 81 boxes resulting in a estimated fractal dimension of 1.77.

Figure 4.1 ***Estimating the fractal dimension of polylines patterns with Box-Counting.***

Although the concept of calculating the Box-Counting dimension is simple, its implementation is not that easy and the results of the estimated fractal dimension can differ due to user-made choices. The steps in Box-Counting are converting data, counting boxes and estimating the fractal dimension shown in Figure 4.2. Crucial choices in application of this method concern the first and third step. I demonstrate each of these steps with the Koch Curve. For all calculations I used Matlab version 2012b.

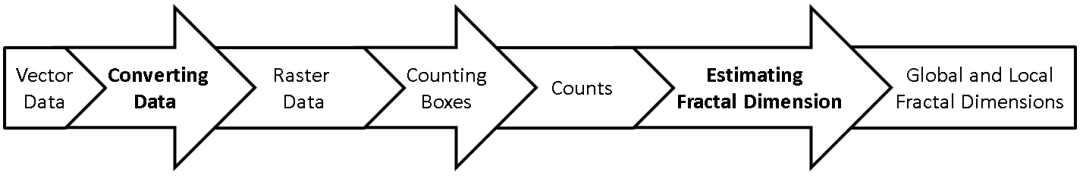


Figure 4.2 ***Three necessary steps for applying Box-Counting to polylines.***

4.1 Converting Vector to Raster Data

The Koch Curve and fault lines can be represented as polylines. A polyline is a connected series of line segments and can be represented by the start and end points of each segment. These points of the polyline segments are defined by Cartesian coordinates. The (*X*, *Y*) coordinates of the start and end point together give information about the magnitude and the direction of a line segment - this is called *vector* format and illustrated in Figure 4.3. The original vector format of the Koch Curve and the fault lines data is used in this study.

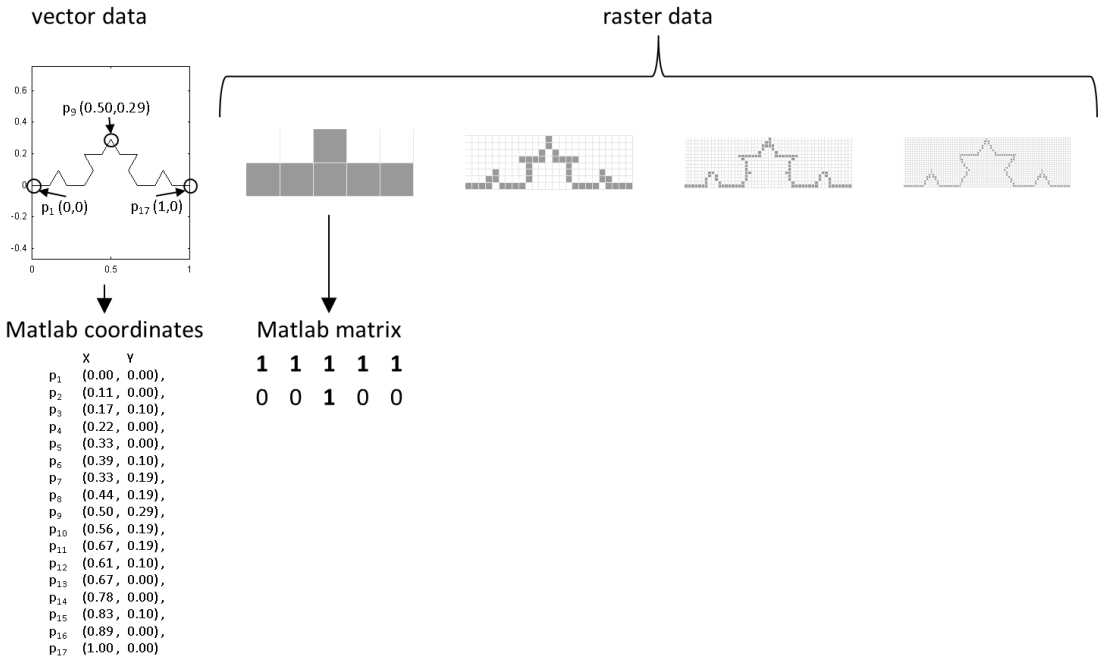


Figure 4.3 ***Presentation of polylines in vector and raster form.*** *Line segments defined by Cartesian coordinates are in vector format. In contrast to this, line segments defined by adjacent boxes are in raster format. A Matlab matrix representing the raster can be thought to be upside-down.*

In contrast to vector format, a line segment can be represented by a series of adjacent square or rectangular boxes. This representation is called *raster* format. These boxes are organized into rows and columns. Each box contains a value representing information. The information indicates for the Koch Curve whether the box is occupied by at least one of its line segments or not. The smaller the boxes of the raster the more detailed and recognizable the Koch Curve in Figure 4.3. Rasters are the basis for Box-Counting.

To use the Box-Counting method the vector data has to be converted to raster data. Here, the steps for constructing and rastering the Koch Curve are shown, starting with the construction of the Koch Curve Vector.

Constructing Koch Curve Vector. The initiator of the Koch Curve is a straight line segment of the length one unit. In the first iteration, one third of the original line segment is taken from the left side, then the direction is changed by +60° (positive meaning anti-clockwise rotation) for continuing with the length of one one third, then the direction is changed by -120° (negative meaning clockwise rotation) for continuing again with the length of one third, then the direction is changed by +60° and the fourth part of the polyline is drawn as shown in Figure 4.4. Each line segment is described by a vector. The length of the line segment is the magnitude and the angle is the direction of the vector. The combination of the three angles, *+60°, -120°, +60°*, is important for further iterations.

Note 0° for the starting point. The four line segments scaled down by $\frac{1}{3}$ are the Koch Curve in the first iteration and this pattern is the generator for further iterations.

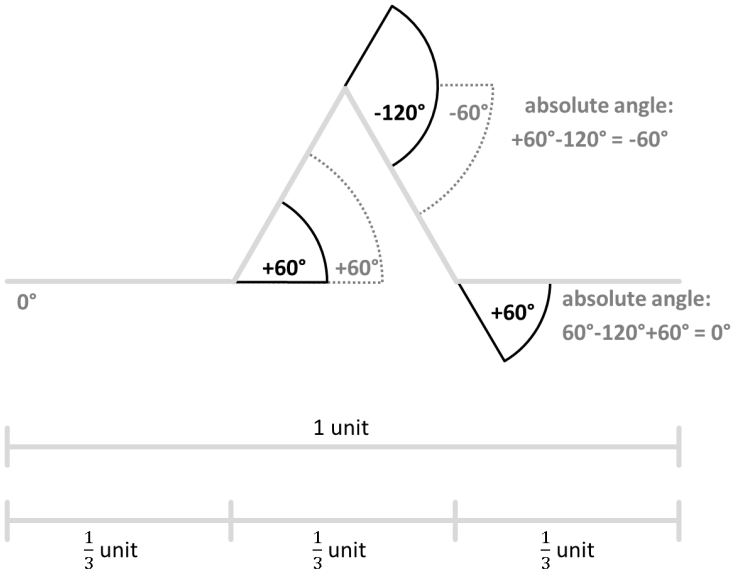


Figure 4.4 ***The generator for constructing the Koch Curve.*** *The generator for the Koch Curve is the pattern of the first iteration. In further iterations straight line segments are replaced by a scaled-down version of the generator. The generator can be represented by the combination of the angles +60° -120° +60° .*

In the second iteration, the generator replaces each of the four scaled down line segments with the generator. The combination of the three angles, *+60°, -120°, +60°*, is inserted after each of the four angles of the first iteration shown in Figure 4.5. For the third iteration the same combination of three angles is inserted again after all 16 angles of the second iteration. The angle of each line segment describes the direction relative to the preceding line segment. The calculation of the Cartesian coordinates based on these relative angles is explained in detail in Appendix A.1.

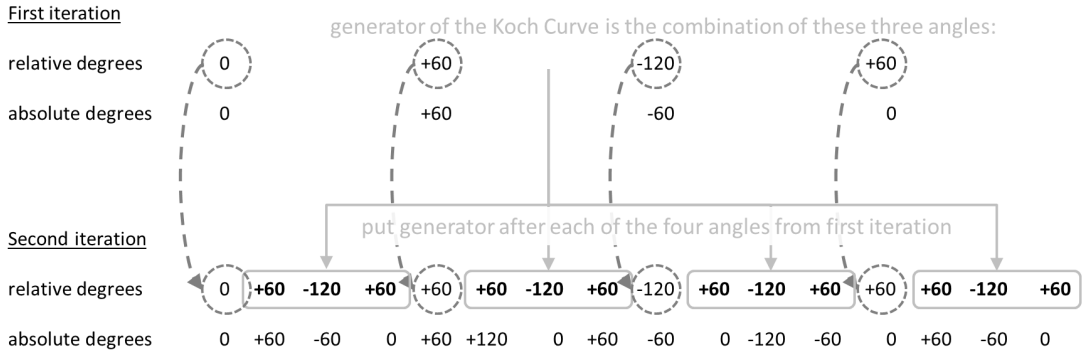


Figure 4.5 ***Creating a Koch Curve in the second iteration.*** *The Koch Curve in the first iteration is based on the start point noted as 0° and the generator represented as the combination of the three angles +60° -120° +60° . For the Koch Curve in the second iteration, the three angles from the generator are placed after each relative angle from the first iteration. These absolute angles in combination of the line segment length in each iteration can be used to calculate the Cartesian Coordinates for each Koch Curve.*

Constructing Raster. The vector format with the Cartesian Coordinates for the Koch Curve needs to be transferred to raster format with boxes. A raster represents this vector data with boxes in rows and columns. The main steps are creating a grid with empty boxes, transforming the Cartesian Coordinates into raster coordinates and encoding the raster coordinates into the grid. The complete process is explained in detail in Appendix A.2.

With the rasters of the Koch Curve in different iterations in Figure 4.6, it looks as if the 17 points from the second iteration are taken over into the third generation, and as if for each line segment 3 additional points are added. We know that this is not the case. The points for the Koch Curve vector are always newly calculated starting from the left and ending on the right side (described in detail in Appendix A.1. This process can be responsible for small rounding differences in the coordinate values. This has the effect that boxes we would expect to be occupied in all iterations are only occupied in some of them, and similarly some boxes are occupied when we would not expect it. This is counter-intuitive. We can see this clearly in Figure 4.6, where the red box is only occupied in the fourth iteration and the blue only in the second iteration.

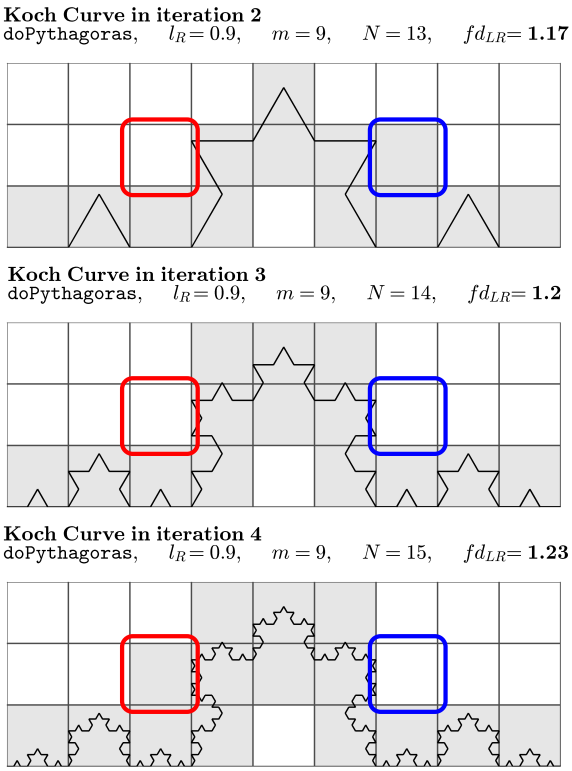
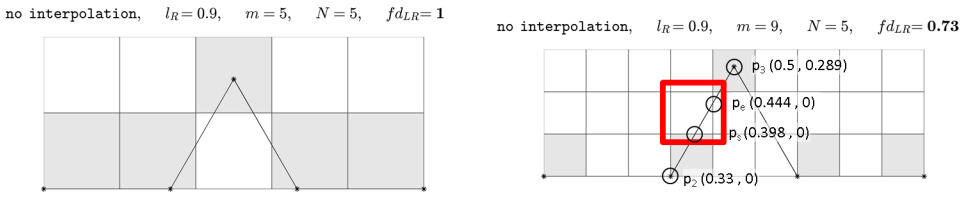


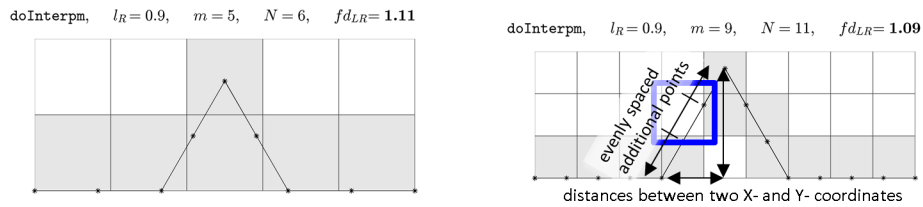
Figure 4.6 ***Influence of rounding differences between Koch Curve iterations on the estimated fractal dimension fd_{LR} at m = 9.*** *Small rounding differences in the Koch Curve constructing have the effect that some boxes are only occupied in the first and third iteration.*

Now, the focus is on the user choices for this conversion from vector to raster form: interpolation, magnification factor, shape of boxes, and the position of the vector data on the grid. The term *grid* is used for the initial pattern of connected empty boxes and the term *raster* after attributing 1 for *occupied* and 0 for *unoccupied* to each box.

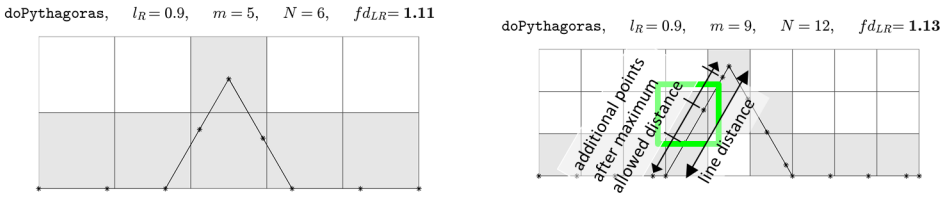
Interpolation. The process of interpolation is essential when points defining the polyline are not in adjacent boxes. Interpolation is the process of adding extra coordinates between the start and end point if the distance between the points exceeds a given maximum length. The procedure is demonstrated with a Koch Curve in the first iteration and $m = 9$ as illustrated in Figure 4.7 (right).



(a) `no interpolation` *The original five points of the first iteration of the Koch Curve are as many as we need to colour the raster at m = 5 appropriate. At m = 9 many boxes a line segment travels through are not identified. To get more boxes identified, we have to interpolate the line, meaning to add more points to it.*



(b) *The Matlab build-in function* `doInterpm` *computes differences between the two X values and two Y values for two points. If the maximum of these two distances exceeds the maximum allowed distance, the line segment between the points is filled with evenly spaced additional points. On the left side one point and on the right two additional points were added*



(c) *The function* `doPythagoras` *focuses not on the distance between two X and two Y coordinates but on the exact length of the line. Additional points are put in consecutively after the maximum allowed distance. This allows identifying all boxes that occupy a line segment equal or longer than the maximum allowed distance.*

Figure 4.7 ***The interpolation methods*** `doInterpm` ***and*** `doPythagoras` ***for the Koch Curve in the first iteration.***

The box edge length e of the grid is calculated by

$$e = \frac{1}{m} = \frac{1}{9} = 0.111 \text{ units} .$$

The interpolation length in units l_U is adjusted to the box edge length of the grid by the relative interpolation length l_R. In the example, the interpolation length l_U for $l_R = 0.90$ is

$$\begin{aligned} l_U &= e \times l_R \\ &= \frac{1}{9} \times \frac{90}{100} = \frac{1}{10} = 0.1 \text{ units} . \end{aligned}$$

The Matlab built-in function `doInterpm` computes the difference between the two X values and the difference between the two Y values of the points for each line segment. If the maximum of these two distances exceeds the interpolation length in units l_U, the line segment between the points is filled with evenly spaced additional points (in more detail in Appendix A.3. To illustrate this clearly, Figure 4.7b shows that adding these additional points does not start at point x_2,y_2 and attaches point for point after the interpolation length l_U. Therefore boxes can be missed such as the blue box.

This is in conflict with the expectation that with an interpolation length of 0.90 of the box edge length every box a line segment travels through would be identified as occupied. Another disadvantage is that `doInterpm` does not focus on the length of a line segment but, rather, the distances between its coordinate values.

Because of these limitations, a new function called `doPythagoras` was written. First, instead of checking the distances between the (X, Y) coordinates, the length of the line segment itself is determined and then interpolated if it is longer than l_U, in our case 0.1 units. Second, new points are added consecutively after the interpolation length l_U is reached. Therefore the points are not evenly spaced. Note, the last part of the line segment can be much smaller than l_U. Figure 4.7c shows how the three points are consecutively added and that the last part of the line segment is smaller. `doPythagoras` is explained in more detail in Appendix A.3.

In contrast to `doInterpm`, the new function `doPythagoras` identified the green box as occupied shown in Figure 4.7c (right). This approach, focusing on the line segment itself and adding the additional points not evenly spaced but after the maximum allowed distance will identify all boxes that occupy a section of a line segment that is equal or longer than l_U.

The more points that are added, the smaller the distance between them and the lower the chance of failing to detect a box that is crossed by a line. This means that a very short line segment crossing a box can cause this single box to be regarded as occupied. In Figure 4.8 some of the identified boxes with `doInterpm` and `doPythagoras` of the raster with $m = 20$ for the Koch Curve in the second iteration occupy very short line segments indicated with red squares. This problem can be mitigated by determining a minimum line segment length. For each box, the length of the line segment travelling through the box needs to be calculated. The process is explained in Appendix A.3. For a small number of boxes this may take a reasonable time, but with increase of the magnification factor the number of boxes increases and then computational power may not be sufficient.

Different relative interpolation lengths l_R influence the number of occupied boxes and the estimated fractal dimension. In Figure 4.9 the number of identified boxes N and the estimated fractal dimension fd_{LR} of rasters with $m = 9$, 15, 25 and 50 for the Koch Curve in the second iteration is calculated with `doPythagoras` for $l_R = 0.3$ equal or higher than for $l_R = 0.9$. The range between the estimated fractal dimension for the two interpolation thresholds increases with the increase of the magnification factor.

In summary, the two methods `doInterpm` and `doPythagoras`, and different relative interpolation lengths l_R affect the estimated fractal dimension.

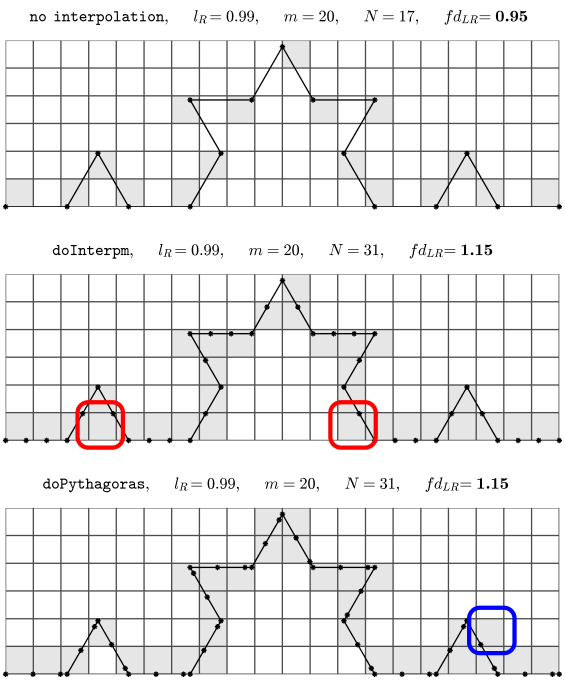


Figure 4.8 ***Influence of interpolation with l_R = 0.99 on the estimated fractal dimension fd_{LR} at m = 20.*** *For the Koch Curve in the second iteration both interpolation methods* `doInterpm` *and* `doPythagoras` *are prone to identify boxes as occupied even though the line segment crossing it is very short.* `doInterpm` *identified the red boxes four and thirteen from the left.* `doPythagoras` *identified the blue box three from the right two up.*

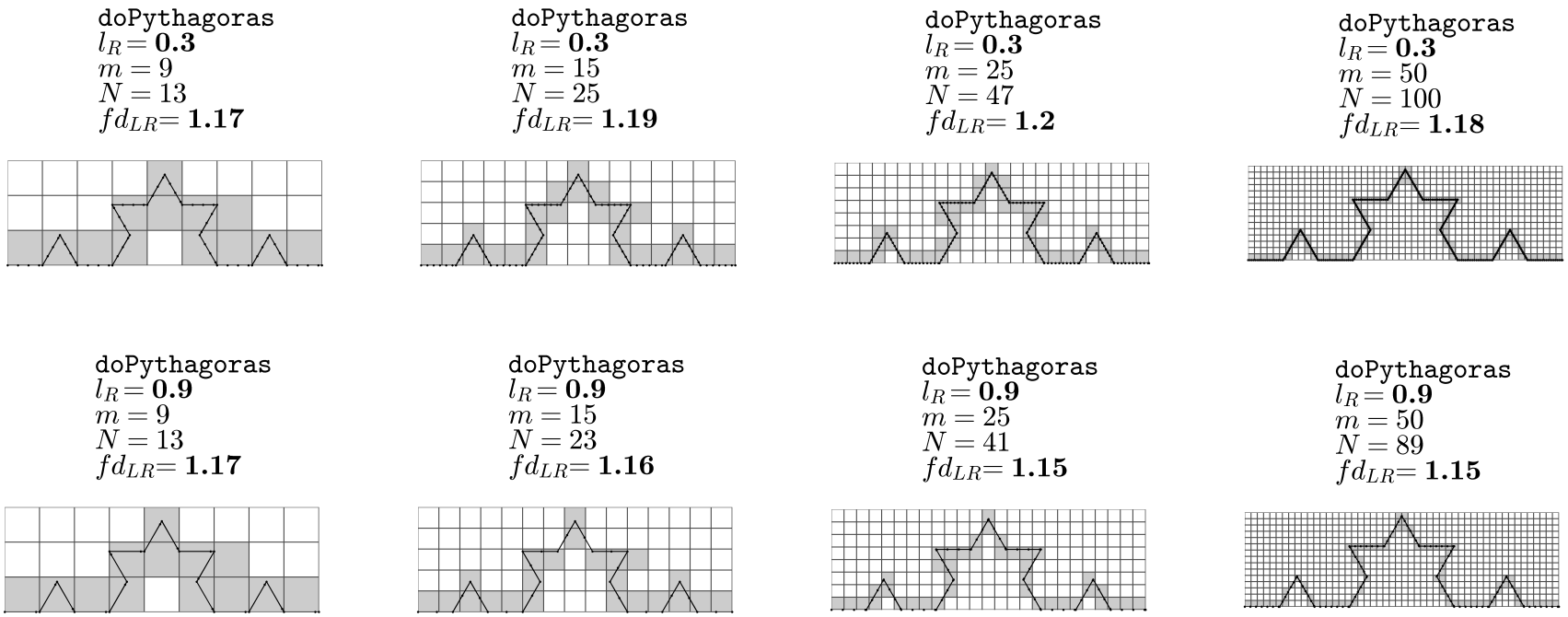


Figure 4.9 ***Influence of different relative interpolation length l_R and magnification factors m on estimated fractal dimension fd_{LR}.*** *With the decrease of the interpolation length l_R, the number of occupied boxes N increases, and thus the estimated fractal dimension fd_{LR} increases. Using the interpolation method* `doPythagoras` *at magnification factor m =50, fd_{LR} at l_R = 0.3 is 0.03 higher than at l_R = 0.9.*
top: *Using l_R = 0.3, fd_{LR} fluctuates at the magnification factors m = 9, 15, 25 and 50.*
bottom: *In contrast, using l_R = 0.9, fd_{LR} decreases with increase of m.*

Magnification Factor. With increase of the magnification factor m, the box size decreases. A raster with a higher magnification factor presents more detail than raster with a low one. For rasters with magnification factor $m = 9$, `doPythagoras` and $l_R = 0.3$ the estimated fractal dimension of the Koch Curve in the second iteration is $fd_{LR} = 1.17$ illustrated in Figure 4.9. With increase of m from 9 to 50 the number of occupied boxes N increases, in contrast, the estimated fractal dimension fluctuates between $fd_{LR} = 1.17$ and $fd_{LR} = 1.2$. For rasters with the same magnification factors but $l_R = 0.9$ the estimated fractal dimension decreases steadily with the increase of the magnification factor from $fd_{LR} = 1.17$ to $fd_{LR} = 1.13$.
This clearly shows that the relative interpolation length interacts with the magnification factor. This result can be explained in two different ways. Either the number of points added with a $l_R = 0.9$ is not high enough to keep the estimated fractal dimension constant across the scales or the estimated fractal dimension is decreasing with the scaling process.

Shape of Boxes The calculation of Box-Counting fractal dimension is based on counting the number of boxes needed to cover the complete pattern on different scales. So far, the boxes were assumed to be square. This approach is found in Matlab functions from Moisy[68] and Anoop[15]. In contrast the Matlab script written by French[33] uses rectangles. The shape of boxes, either squares or rectangles, can affect the estimated fractal dimension.

a) Squares. There are two ways of constructing a raster with square boxes. One way is to construct *multiple raster*, one for each magnification factor (explained in detail in Appendix A.2. The other way is to base the rasters with smaller magnification factor on the raster with the biggest magnification factor called *single raster*. The second way uses Moisy's[68] and Anoop's[15] Matlab functions.

Using a single raster, Box-Counting is based on a Matlab matrix that reflects the raster with the smallest box edge length e, or with the highest magnification factor m. This is called the raster of the first generation, $g = 1$. The raster box of the smallest size is represented by one matrix cell, mathematically defined to be 2^0 matrix cells.

As an example, rasters are created for the Koch Curve in the second iteration as illustrated in Figure 4.10. I start for the raster in the first generation $g = 1$ with the highest magnification factor, e.g. $m_g = 2^7$. The length of the box e_g for this raster is $e_1 = \frac{1}{2^7}$ units. The box is represented by one matrix cell (2^0). In the following generations of raster the length of boxes doubles, beginning in the second generation with

$$\begin{aligned} e_2 &= 2^1 \times \frac{1}{2^7} = \frac{1}{2^6}, \\ \text{and in general} e_g &= 2^{g-1} \times e_1 . \end{aligned}$$

The magnification factor m halves, here $m_2 = 2^6$ and the number of matrix cells representing the second smallest box is $2^1 \times 2^1$, and in general $2^{g-1} \times 2^{g-1}$.

The generation of rasters ends at $m = 2^0$, a box edge length $e = \frac{1}{2^0}$ and the box size is represented by a $2^g \times 2^g$ matrix. For this reason Moisy's function[1] increases the number of rows and columns in the matrix to the size $2^n \times 2^n$ if the width and the height of the matrix representing the object are not equal or if they are not of the size 2^n illustrated in Figure 4.11. Empty cells in the increased matrix are filled up with zeros. This can cause an extreme change to the size of the matrix.

Moisy[69] bases the magnification factor on the artificial increased matrix. This causes a different size of box edge length in units and can lead to a different estimated fractal dimension, e.g. for the Koch Curve in the second iteration in Figure 4.11 the estimated fractal dimension for $N = 3$ boxes at the magnification factor $m = 3$ based on

[1] Anoop[15] mentions that their function is for a 2^9 matrix and so, there is no increase necessary

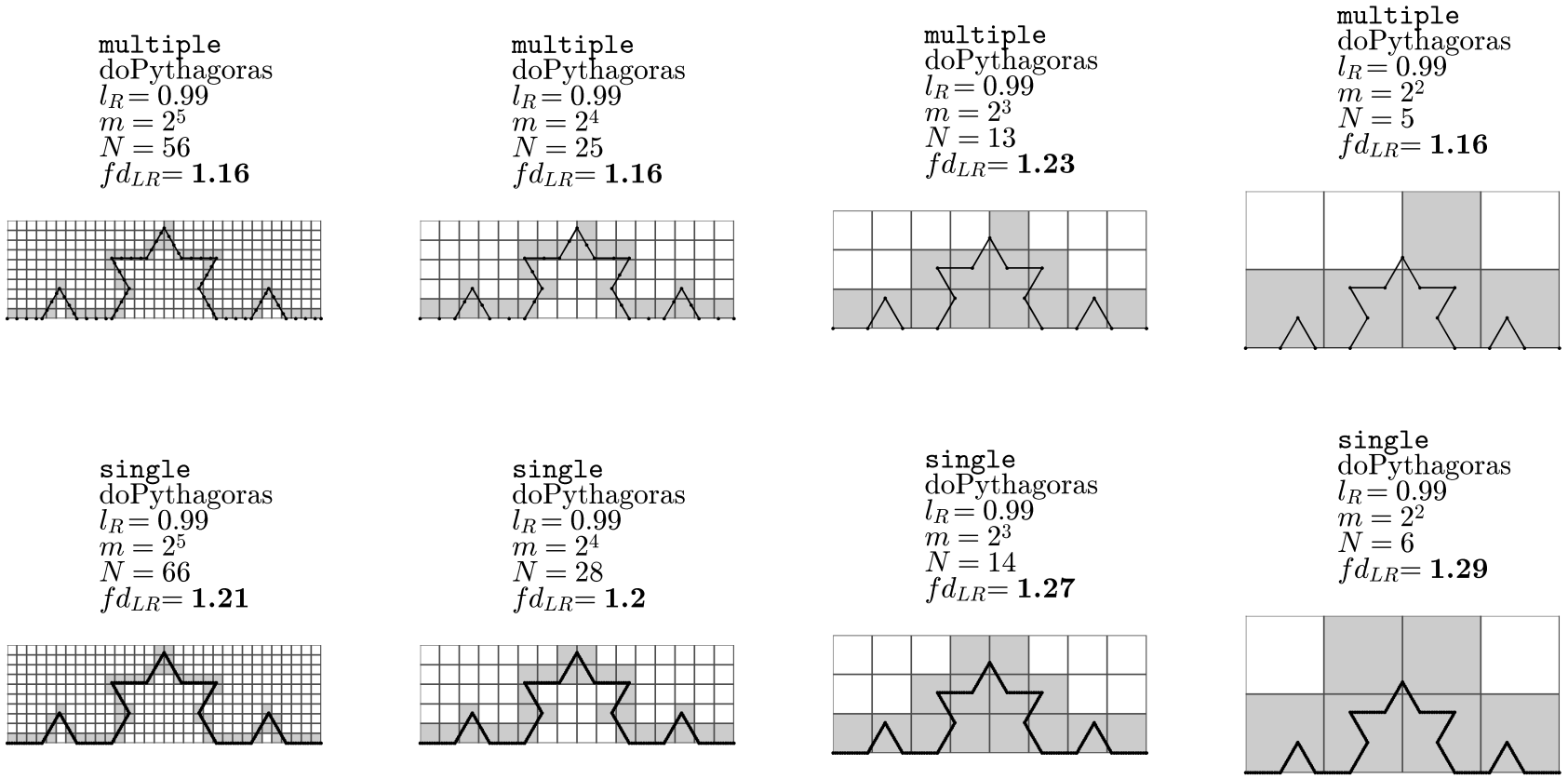


Figure 4.10 ***Influence of single vs multiple matrices on the estimated fractal dimension fd_{LR} at $2^2 \geq m \geq 2^7$.*** *The matrix for the magnification factor 2^7 can be used for smaller magnification factors such as 2^6, 2^5, .. 2^0. Using the matrix for the highest magnification factor for the smaller magnification factors leads to a higher number of occupied boxes than using multiple matrices. At magnification factor 2^5, there are N = 56 occupied boxes using a separate matrix, and N = 66 based on the matrix for magnification factor 2^7. This is an increase of more than 17 %.*

the original size is

$$fd_{LR} = \frac{log(N)}{log(m)} = \frac{log(3)}{log(3)} = 1 \ .$$

and the estimated fractal dimension for N = 3 boxes at the magnification factor $m = 2^2$ based on the artificial increased matrix of the size 2^n is

$$fd_{LR} = \frac{log(N)}{log(m)} = \frac{log(3)}{log(2^2)} \quad = \quad 0.79 \ .$$

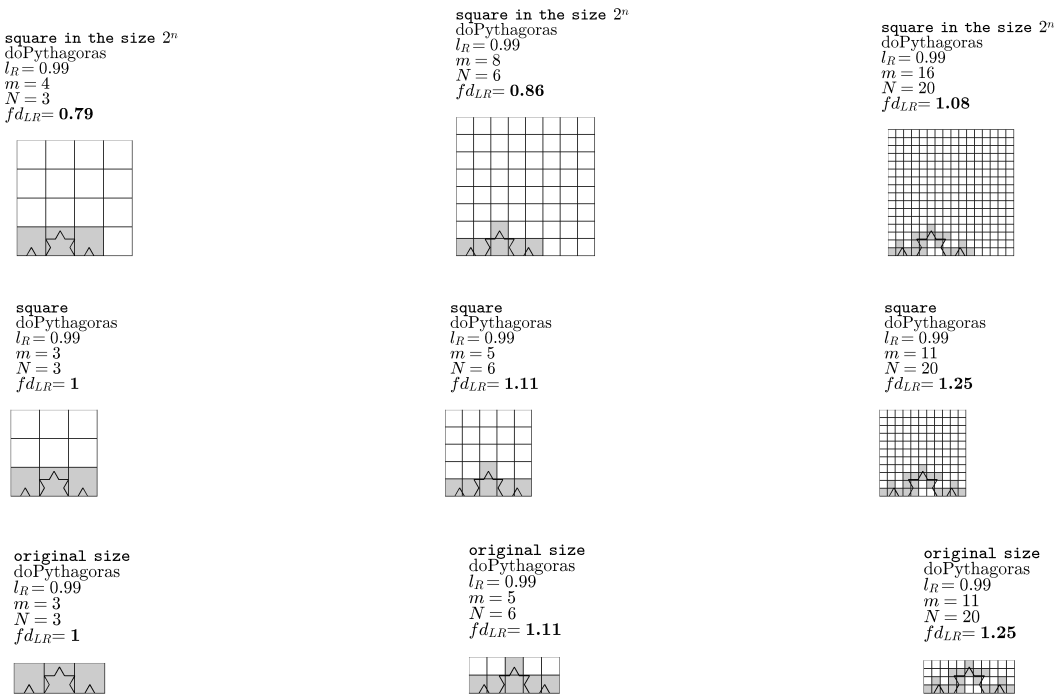


Figure 4.11 ***Difference in basing magnification factor m on actual pattern or artificial increased matrix.*** *Taking the magnification factor m = 4 (2^2) instead of m = 3 causes a change of the estimated fractal dimension from fd_{LR} = 1 to fd_{LR} = 0.79. Taking m = 8 (2^3) instead of m = 5 causes a change of from fd_{LR} = 1.11 to fd_{LR} = 0.86.*

At first sight another disadvantage of Moisy's and Anoop's functions is that the steps between magnification factors are not equal. The doubling of the box edge length between generations of rasters causes a two fold increase in the intervals between the magnification factors. Taking the log of the magnification factors equals the interval between two adjacent magnification factors.

The further disadvantage of using a single matrix is due to small line segments that just cross the smallest box. Interpolation related to the box edge length may add a point in the smallest box and make it occupied. If this matrix is kept for other magnification factors such as 2^{n-1}, 2^{n-2}, ..., 2^1, four smaller boxes are combined into one bigger box. Each box, independent of whether it is a small or big box containing the smallest box with the short line, would be characterized by this short line. Keeping the same matrix always leads to more occupied boxes than multiple matrices with an interpolation length appropriate to each box size.

With the second iteration of the Koch Curve using multiple matrices, the number of occupied boxes is N = 56 at $m = 2^5$ and N = 25 at $m = 2^4$ illustrated in Figure 4.10. Using a single matrix based on the magnification factor 2^7, the number of occupied boxes is higher (N = 120 at $m = 2^6$ and N = 66 at $m = 2^5$. For these examples, the increase of occupied boxes between multiple matrices and single matrix is more than 11 %. This results in a difference of 0.05 in fd_{LR} at $m = 2^5$.

These disadvantages are not outweighed by the advantage of faster computation in using only one single matrix for all magnification factors. Therefore it is suggested to base the raster on multiple Matlab matrices, one for each magnification factor.

b) Rectangles. French[33] uses rectangles for the shape of the box, starting with the complete pattern as the largest rectangular box. The process iteratively decreases the box size by increasing the number of boxes, starting with 1^2, 2^2, ... , m^2 boxes, where m is the magnification factor. In contrast to Moisy's[68] and Anoop's[15] function for square boxes, in French's function the biggest rectangle frames the pattern and therefore no additional space is added.

The steps between the magnification factors are equal. However, if a logarithmic transformation is used then the steps in the log of the magnifications factor will be uneven. This skew causes the log of smaller magnification factors to be further from the mean compared with the log of higher magnification factors. For example,

for the magnification factors	2	3	4	5	6	7	8	9
the natural log of the magnification factors are	0.69	1.09	1.38	1.60	1.79	1.94	2.07	2.19

and the mean is 1.60. The log of the smallest magnification factor $m = 2$ is further from the mean than the log of $m = 9$. When a regression line is fitted to the log of magnification factors and the log of the number of occupied boxes, the log of magnification factors further from the mean has a greater potential to influence the slope. This is important for calculating the estimated global fractal dimension fd_{GS} as the slope of a regression line described in Section 4.2. The amount of influence each box size has is known as its *leverage* value. In this example, the magnification factor $m = 2$ would have the biggest leverage.

The area of rectangular boxes is generally smaller than that of square boxes at the same magnification factor m. One reason is that the biggest box for the rectangle is based on the actual pattern whereas the height and width using squares needs to be increased to a multiple of the box edge length (described in Appendix A.2). Another reason is that the number of rows and columns are equal whereas these number can differ using squares. For example, using a magnification factor $m = 5$ for the second iteration of the Koch Curve, there are five columns and five rows for the rectangle method. In contrast to this, there are five columns but only two rows for the square method illustrated in Figure 4.12. The difference in the size of the boxes is compensated by the number of boxes. The estimated fractal dimension using smaller rectangular boxes is always higher than using square boxes. For example, for the Koch Curve in the second iteration at $m = 7$, the estimated fractal dimension $fd_{LR} = 1.36$ using rectangular boxes is higher than $fd_{LR} = 1.18$ using square boxes.

Position of Vector on Grid. Another factor that has an influence on the estimated fractal dimension is the position of the origin and the orientation of the grid relative to the Koch Curve in vector format. The relative position is changed by translation of the Koch Curve, and the orientation is changed by rotation. The location and the orientation of the pattern on the grid is essentially a random choice by the user.

a) Translation. Intuitively, the Koch Curve is aligned to the upper-left or lower-left corner of the empty grid, but sliding the pattern either right-left or up-down can change the number of boxes needed to cover the pattern. A simplified example is shown in Figure 4.13, which has two horizontal lines. With the translation the number of occupied boxes changes from $N = 7$ to $N = 8$. With this increase, the estimated fractal dimension increases from $fd_{LR} = 1$ to $fd_{LR} = 1.15$. For each box size the translation up to one box edge length can result in a different number of occupied boxes. For grids with bigger boxes there are either more translations or the steps between the different positions are bigger.

To illustrate this, the vector of the Koch Curve is translated in the second iteration on a grid with square boxes and $m = 9$ five times horizontally and five times vertically, in total 25 different placements. The relative

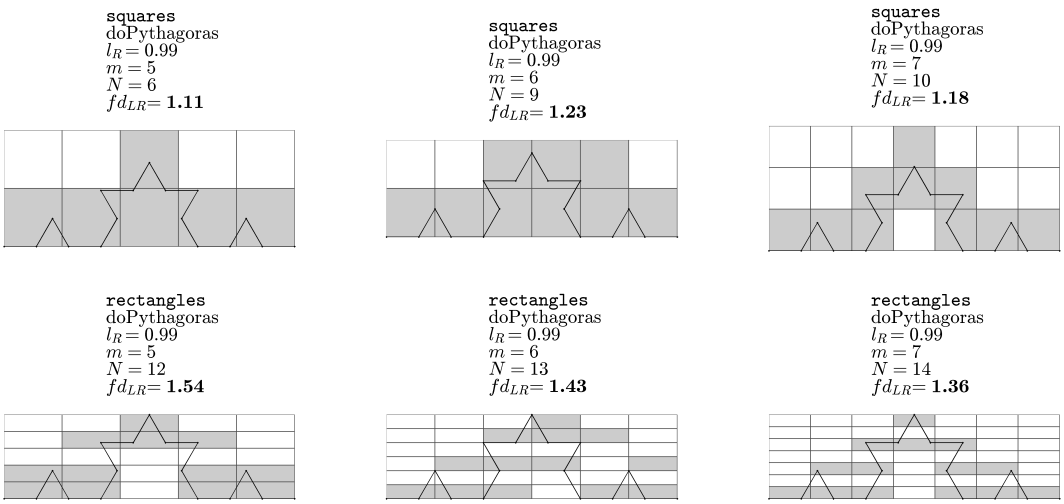


Figure 4.12 ***Different shapes of boxes - rectangles and squares.*** *The difference between two successive box sizes is smaller with rectangles than with squares. The reason is that the biggest box for the rectangle is based on the actual pattern whereas the height and width of boxes using squares need to be increased to a multiple of the box edge length. In general, the areas of rectangles are always smaller than those of squares at the same magnification factor.*

steps in horizontal direction h_R and vertical direction v_R were 0.0, 0.1, 0.2, 0.3, 0.4 of the box edge length, see Appendix B.1. For these 25 positions, the minimum number of occupied boxes was N = 13 at the original position, h_R = 0.0 and v_R = 0.0, and the maximum was N = 16 at h_R = 0.0 and v_R = 0.4. This is an increase of 23 percent. Figure 4.14 gives the impression that the vector of the position h_R = 0.0 was aligned at the bottom of the grid, and the vector of the position h_R = 0.4 was aligned at the top. The range in the estimated fractal dimension fd_{LR} is 0.09 between 1.17 and 1.26. This shows that even that the number of the grid boxes were the same, in both cases 3×9, the number of occupied boxes and the estimated fractal dimension can differ.

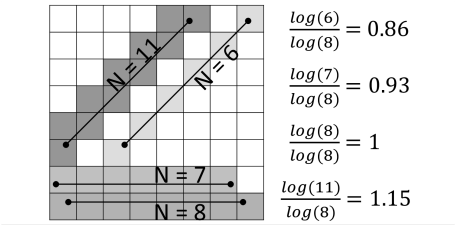


Figure 4.13 ***Simplified illustration for influence of translation and rotation of a straight line segment on the estimated fractal dimension.*** *Four line segments with the same lengths showing how the orientation of the line segment influences not only the number of boxes needed to cover the line segments but also the estimated fractal dimension. The number of occupied boxes ranges from N = 6 to N = 11 and the estimated fractal dimension range between fd_{LR} = 0.86 and fd_{LR} = 1.15.*

Figure 4.14 ***Influence of translation of the vector on the grid using square boxes on the estimated fractal dimension.*** *Moving the vector on the grid vertically and horizontally can change the number of occupied boxes N and leads to different estimated fractal dimensions fd_{LR}. Here, the minimum is fd_{LR} = 1.17 when the Koch Curve is aligned at the lower left. Moving the Koch Curve upwards by $\frac{4}{10}$ of the box edge length results in a change of the estimated fractal dimension of 0.09.*

b) Rotation. The orientation of the Koch Curve can be in any arbitrary direction. Very often, for landscape patterns the geographic direction is used. For other patterns, the real world orientation may be used or just the orientation the image was taken with. However, Box-Counting is not based on any built-in orientation, and rotating the pattern has an impact on the estimated fractal dimension. For example, in Box-Counting, the dimension of a line segment depends on its orientation. As a simplified example, in Figure 4.13 line segments of the same length but different orientations cover a different number of boxes and this results in a range in the estimated fractal dimension from fd_{LR} = 0.86 to fd_{LR} = 1.15.

Not only does the direction of a single line segment matter, but the complete pattern can result in differences. For the Koch Curve in the fourth iteration at m = 20, the rotation of the vector on a grid with square boxes in steps of 5° from 0° to 90° leads to different metric size of the raster illustrated in Figure 4.15 and Appendix B.2. At m = 20 the minimum number of occupied is N = 45 at angle 0° and the maximum is N = 59 at 25° and 65°. The difference of 14 boxes leads to a range in the estimated fractal dimension of 0.09 between fd_{LR} = 1.27 and fd_{LR} = 1.36.

Using the rectangle method for the same iteration of Koch Curve and m = 20, the orientation of the Koch Curve notably influences the metric size of the rectangular boxes as illustrated in Appendix B.3. The minimum is found

at 60°with fd_{LR} = 1.43 and the maximum at 0°with fd_{LR} 1.56 shown in Figure 4.16. Not only is the estimated fractal dimension with rectangular boxes higher than with square boxes, but the range in the estimated fractal dimension of 0.13 between the 19 rotation positions is higher than 0.09 using square boxes.

Figure 4.15 ***Influence of rotation of the vector on the grid using square boxes on the estimated fractal dimension.*** *The rotation of the fifth Koch curve iteration on a grid with magnification factor m = 20 results in a difference in the estimated fractal dimension of 0.09 between fd_{LR} = 1.27 at the orientation 0° and fd_{LR} = 1.33 at 60°.*

Figure 4.16 ***Influence of rotation of the vector on the grid using rectangular boxes on the estimated fractal dimension.*** *The rotation of the vector using rectangular raster boxes results not only in different estimated fractal dimension but also different metric sizes of the boxes. The range in the estimated fractal dimension at m = 20 is from fd_{LR} =1.43 at 60° and fd_{LR} =1.56 at 0°.*

4.2 Calculating Fractal Dimension

The final step of Box-Counting is the calculation of the fractal dimension. Although all methods agree in that they compare the relationship between number of occupied boxes and box edge size on different scales, they differ in the mathematical calculation. An estimated local fractal dimension is specified to individual box edge sizes. If an estimated global fractal dimension can describe an object for all scales the pattern is called monofractal. If there is a change in the estimated fractal dimensions for different scale ranges, the object is called band-limited.

I illustrate the calculation of three local fractal dimensions and two global fractal dimension with the fifth iteration of the Koch Curve illustrated in Figure 4.17 and raster of 10 magnification factors from 10, 20, 30, ..., 100. The Koch Curve was interpolated with `doPythagoras` and an interpolation length of $l_R = 0.99$.

Local Fractal Dimension. Three methods are chosen to illustrate the difference between local methods. Two methods are variations of Richardson's slope expression explained in Section 3.1. The third method is the calculation of the relationship between the log of the number of occupied boxes and the log of the magnification factor at each scale as illustrated in Figure 3.5 for the Koch Curve and Sierpinski carpet and already used as fd_{LR} in this chapter.

The two slope methods are based on the gradient of points representing the magnification factor and their corresponding number of boxes needed to cover the Koch curve object in a log-log coordinate system. This system uses logarithmic scales on both the horizontal and vertical axes. Moisy's[69] calculation of the slope is calculated with forward differential coefficients between two adjacent points shown in Figure 4.18. The slope of a line fitted to the point p_1 (10,16) indicating $N = 16$ at $m_1 = 10$ and point p_2(20,44) indicating $N = 44$ at $m_1 = 20$ is

$$
\begin{aligned}
fd_{FD([i],[i]+1)} &= \frac{log(N_{[i]+1}) - log(N_{[i]})}{log(m_{[i]+1}) - log(m_{[i]})} \\
fd_{FD(m_1,\, m_2)} &= \frac{log(N_{m_2}) - log(N_{m_1})}{log(m_2) - log(m_1)} \\
fd_{FD(10,\, 20)} &= \frac{log(44) - log(16)}{log(20) - log(10)} = 1.46\,,
\end{aligned}
$$

where $fd_{FD[i],[i]+1)}$ is the estimated local fractal dimension based on forward differences at the $[i]$th sorted magnification factor m, and N is the number of occupied boxes. The log base for the number of occupied boxes and the magnification factor should be the same. The number of resulting estimated fractal dimension values is one less than the number of magnification factors.

In contrast to that method, Moisy(2008)[68] uses in a different function a mix of forward, central and backward differential coefficients. Forward differences are only used for the smallest magnification factor so that

$$
\begin{aligned}
fd_{MD([i])} &= \frac{log(N_{[i]+1}) - log(N_{[i]})}{log(m_{[i]+1}) - log(m_{[i]})} \\
fd_{MD(m_1)} &= \frac{log(N_{m_2}) - log(N_{m_1})}{log(m_2) - log(m_1)} \\
fd_{MD(10)} &= \frac{log(44) - log(16)}{log(20) - log(10)} \quad = \quad 1.46\,,
\end{aligned}
$$

where fd_{MD} is the estimated local fractal dimension based on a mix of forward, central and backward differences. Backward differences are used for the highest magnification factor so that the slope for the points p_9 (90,308) and

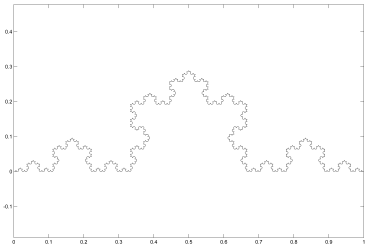


Figure 4.17 ***Koch Curve in the fifth iteration used for estimating the local and global fractal dimensions.*** *This pattern is used for the comparison of three local and two global fractal dimensions*

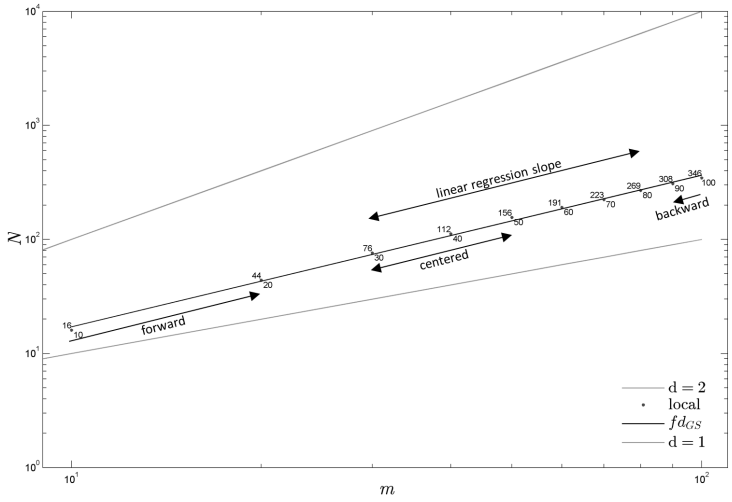


Figure 4.18 ***Estimated global fractal dimension for the Koch Curve in the fifth iteration.*** *The lower integers are the magnification factors from m = 10. 20, 30, ..., 100, the upper integers are the number of occupied boxes N. The estimated local fractal dimension is calculated in three ways: fd_{FD} with forward differences , fd_{MD} with a mixture between forward, central and backward differences and fd_{LR} with the log ratio between the N and m. The slope of the regression line is the global fractal dimension fd_{GS}. The line d = 1 and d = 2 represent dimension 1 and 2, respectively. All of the points for the Koch Curve are between them and therefore their dimension is fractal, between 1 and 2.*

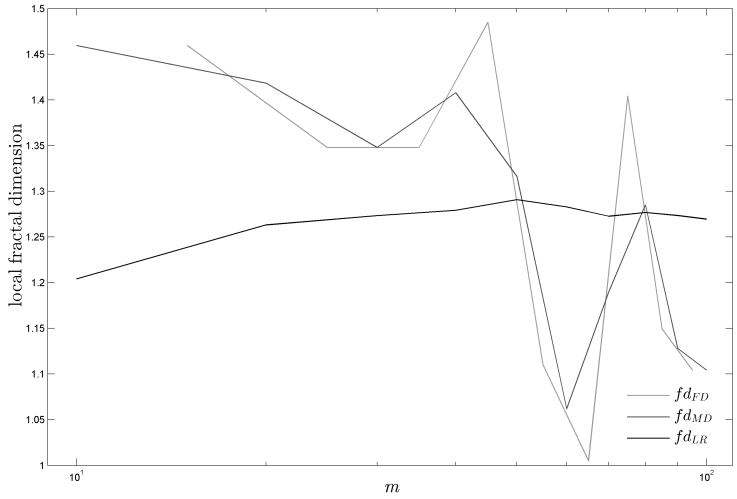


Figure 4.19 ***Estimated local fractal dimensions for the Koch Curve in the fifth iteration.*** *The estimated fractal dimension using the slope methods fd_{FD} and fd_{MD} fluctuates, whereas fd_{LR} is more constant.*

p_{10} (100,346) is

$$
\begin{aligned}
fd_{MD([i])} &= \frac{log(N_{[i]}) - log(N_{[i]-1})}{log(m_{[i]}) - log(m_{[i]-1})} \\
fd_{MD(m_{10})} &= \frac{log(N_{10}) - log(N_9)}{log(m_{10}) - log(m_9)} \\
fd_{MD(100)} &= \frac{log(346) - log(308)}{log(100) - log(90)} \quad = \quad 1.10 \; .
\end{aligned}
$$

Central differences are used for intermediate magnification factors. The estimated fractal dimension at one magnification factor is calculated as the slope between the next lower and next higher magnification factors. For the estimated fractal dimension at m_4 = 40 the adjacent points p_3 (30,76) and p_5 (50,156) are relevant such that

$$
\begin{aligned}
fd_{MD([i])} &= \frac{log(N_{[i]+1}) - log(N_{[i]-1})}{log(m_{[i]+1}) - log(m_{[i]-1})} \\
fd_{MD(m_4)} &= \frac{log(N_5) - log(N_3)}{log(m_5) - log(m_3)} \\
fd_{MD(40)} &= \frac{log(156) \; - \; log(76)}{log(50) - log(30)} \quad = \quad 1.41 \; .
\end{aligned}
$$

The mix between forward, central and backward differential coefficients results in one estimated fractal dimension for each box size. Whereas the first two methods fd_{FD} and fd_{MD} involve neighbored values , the third method fd_{LR} focuses just on one particular box size by using the relationship between the log of occupied boxes N and the log of magnification factor m, e.g. for the estimated fractal dimension at m_{10} = 100

$$
fd_{LR(100)} = \frac{log(346)}{log(100)} \quad = \quad 1.27 \; ,
$$

where fd_{LR} is the estimated local fractal dimension based on the log ratio. This method is the same as used for the Koch curve and Sierpinski carpet illustrated in Figure 3.5 and already used earlier in this chapter.

The estimated fractal dimension using the slope methods fd_{FD} and fd_{MD} fluctuates, whereas fd_{LR} is more constant. The range in the estimated fractal dimension between these three estimated local fractal dimensions is 0.26 at m = 10 shown in Figure 4.19 and Appendix C.1.

Global Fractal Dimension. After the estimation of the local dimension, the estimated global fractal dimension can be determined. This is only sensible if the local dimension show similar values. Similar to the local methods, there appears to be no standard for calculation.

Moisy (2008)[69] selects the estimated local fractal dimension of centered magnification factors and determines their mean. In the example, using the estimated local fractal dimension fd_{LR} and deciding to ignore four magnification factors, the two smallest and two biggest, then

$$
\begin{aligned}
fd_{GM} &= \frac{1}{n-j+1} \quad \sum_{i=j}^{n} fd_{LR}[i] \\
&= \frac{1}{8-3+1} \quad \sum_{i=3}^{8} fd_{LR}[i] \\
&= \frac{1}{6} \qquad\qquad (i_3 \quad + i_4 \quad + i_5 \quad + i_6 \quad + i_7 \quad + i_8) \\
&= \frac{1}{6} \qquad\qquad (1.27 \quad + 1.28 \quad + 1.29 \quad + 1.28 \quad + 1.27 \quad + 1.28) \quad = 1.28 \; .
\end{aligned}
$$

where fd_{GM} is the estimated global fractal dimension based on the mean at the $[i]$th sorted magnification factor m,

j is the index for the first of the centered estimated local fractal dimension, n is the index for the last of the centered estimated local fractal dimension.

In the example, I have the local fd_{LR} for 10 magnification factors. The fd_{LR} for the two biggest and smallest magnification factors are disregarded, so the sum includes the estimated fractal dimensions from the third smallest magnification factor $fd_{LR[3]}$ up to the third biggest magnification factor $fd_{LR[8]}$. There is a difference of 0.02 between fd_{GM} = 1.28 and the real estimated fractal dimension of the Koch Curve with 1.26.

In contrast, Anoop (2011)[15], Costa (2011)[22] and French (2007)[33] determine the estimated global fractal dimension as the slope of a regression line with

$$fd_{GS} = \frac{\sum_{i=1}^{n}(log(m_i) - \overline{log(m)})(log(N_i) - \overline{log(N)})}{\sum_{i=1}^{n}(log(m_i) - \overline{log(m)})^2}$$

$$= \frac{\mathrm{Cov}[log(m), log(N)]}{\mathrm{Var}[log(m)]} = 1.33\ ,$$

where fd_{GS} is the estimated global fractal dimension calculated as the slope of a regression line. Depending on the chosen magnification factor, the *leverage* value can be very different (described in Section 4.1). In the example the determination coefficient is 0.999. This means that the points in Figure 4.18 almost lie on the regression line. This shows that the relationship between occupied boxes and the magnification factor is constant throughout all scales. This is an excellent result, but the real fractal dimension is 1.26, so that there is a difference of 0.07. This difference using fd_{GS} is higher than 0.02 using fd_{GM}. This illustration of different calculations of the estimated fractal dimension clearly shows that not only the pattern itself determines the local estimated fractal dimension but the mathematical method that is applied.

4.3 Summary and Discussion

This research has highlighted the potential differences in estimated fractal dimensions because of user choices. I summarized the most extreme cases for the Koch Curve in Table C.2 by providing the fractal dimension fd_{LR}. For the Koch Curve differences are observed at the vector data to raster data conversion for different interpolations, magnification factors, shape of boxes, and the positions of the vector on the grid.

Magnification factor and interpolation. The choice of the magnification factor and the interpolation of the data lead to different results. The number of added points with interpolation changes with the change of the relative interpolation length l_R, the methods to calculate how many additional points are needed, and if these points are added equally-spaced or consecutively each after the interpolation length.

The interpolation with the Matlab function `doInterpm` and `doPythagoras` lead to an increase in the fd_{LR} up to 0.40 for the first iteration of the Koch Curve at m = 5. (Table C.2 a). For the second iteration of the Koch Curve the estimated fractal dimension with both interpolation methods increases only by 0.20 at m = 20. One reason between the differences in the range of 0.40 and 0.20 is that the original vector for the first iteration only has 5 points, whereas the number of original points for the second iteration is already 17. The number of additional points is for the second iteration lower and therefore the difference between the three results (no interpolation, `doInterpm` and `doPythagoras`) is smaller (Table C.2 b).

In contrast to this big difference, the differences are smaller between different Koch Curve iterations as long as the Koch Curve vector data is interpolated. Comparing the pattern of the third and fourth iteration, we might have expected that exactly the same boxes are occupied. However, small rounding differences lead to a change of 0.07 (Table C.2 c).

Decreasing of the relative interpolation length l_R increases the estimated fractal dimension. For the second

iteration of the Koch Curve and using $l_R = 0.3$ and $l_R = 0.99$, there is a difference of 0.05 in fd_{LR} at $m = 25$ (Table C.2 d).

None of the estimated fractal dimension was close to the true fractal dimension of the Koch Curve ($fd = 1.26$). However, for the purpose of visualization of the method Box-Counting with the Koch Curve only small magnification factors could be used. High magnification factors lead to a high number of boxes, and thus the rasters would not have been presentable. For future research I suggest to use both interpolation methods and different relative interpolation lengths, and to evaluate differences in the estimated fractal dimension at many scales.

Shape of Box. The choice of the box form for the raster, either squares or rectangles, is crucial. For using squares, there is a difference in the result of the estimated fractal dimension between using a single matrix or multiple matrices with adapted interpolation length. The estimated fractal dimension is $fd_{LR} = 1.21$ at $m = 2^5$ based on the single matrix of $m = 2^7$. Creating a new matrix based on a longer, more appropriate interpolation length, the estimated fractal dimension decreases by 0.06 to $fd_{LR} = 1.16$ (Table C.2 e).

When the magnification factor is not based on the actual pattern, but on an artificial increased matrix of the size 2^n, the estimated fractal dimension can drop by 0.25 from $fd_{LR} = 1.11$ to $fd_{LR} = 0.86$ as we see in (Table C.2 f). The box form also influence the result in the fd. The difference in the estimated fractal dimension using squares and rectangles ranges with 0.43 between $fd_{LR} = 1.11$ and $fd_{LR} = 1.54$ at $m = 5$ (Table C.2 g). It is always higher for rectangles than squares as their areas is smaller at the same magnification factor.

Positioning. The translation and rotation of the vector data on the empty grid influence the estimated fractal dimension. In the example, a vertical translation of $v_R = 0.4$ of the box edge length increased the estimated fractal dimension by 0.09 to $fd_{LR} = 1.26$ (Table C.2 h). The positioning has an even greater influence with respect to rotating the vector data on the grid. Rotation on a grid with square boxes results in a difference of 0.10 in fd_{LR} at $m = 20$ (Table C.2 i). The range for rotating on a grid with rectangular boxes at the same magnification factor is with 0.13 even bigger (Table C.2 j).
Following the quote Schumacher[(87)2] that "any intelligent fool can make things ... more complex ... It takes a touch of genius and a lot of courage to move in the opposite direction" I recommend not to target the maximum, but the minimum count. The minimum count will result in a smaller estimated fractal dimension.

Differences in the fractal dimension are also observed at the step of estimating the fractal dimension with different methods.

Local Fractal Dimension. The fractal dimension fd_{FD} and fd_{MD} estimated by the two slope methods fluctuates, whereas the fd_{LR} is more stable.

Whereas the $fd_{LR(m)}$ continuously decreases, the $fd_{FD(m)}$ and $fd_{MD(m)}$ change the direction. For example, the $fd_{MD(50)} = 1.32$ decreases to $fd_{MD(60)} = 1.06$ and increases to $fd_{MD(70)} = 1.19$ listed in Table C.1.

The same is the case for the $fd_{FD(m_1,m_2)}$ between $30 \geq m \geq 70$. There is a increase from $fd_{FD(30,40)} = 1.35$ to $fd_{FD(40,50)} = 1.48$ and a decrease to $fd_{FD(50,60)} = 1.11$. At $m = 60$, there is a difference of 0.22 between fd_{MD} and fd_{LR}. The reason is that the calculation of $fd_{MD}$is focusing on the difference between values instead of focusing on the values themselves. The fd_{MD} is calculated with

$$fd_{MD(60)} = \frac{log(N_{70}) - log(N_{50})}{log(m_{70}) - log(m_{50})}$$
$$fd_{MD(60)} = \frac{log(223) - log(156)}{log(70) - log(50)} = 1.06 .$$

[2]former Chief Statistician for the British Control Commission and Chief Economic Adviser to the National Coal Board

The same estimated fractal dimension of 1.06 can result from an even more complex pattern. For Box-Counting that means more occupied boxes. As an example N = 1,000 boxes at m = 50 and N = 1,430 boxes at m = 70 lead to

$$fd_{MD(60)} = \frac{log(1,430) - log(1,000)}{log(70) - log(50)} \quad = \quad 1.06 \; .$$

This clearly shows that the fd_{FD} and fd_{MD} are only suitable for fractals that at each scale have the exact relationship between *log(N)* and *log(m)*. For example, taking $N = 50^{1.5}$ occupied boxes at m = 50 and $N = 70^{1.5}$ occupied boxes at m = 70, the estimated fractal dimension is

$$fd_{MD(60)} = \frac{log(70^{1.29}) - log(50^{1.29})}{log(70) - log(50)} \quad = \quad 1.29 \; .$$

The relationship between *log(N)* and *log(m)* is not the same as we see from fd_{LR} in Table C.1. Even small differences such as 0.02 between m = 50 and m = 70 cause extreme changes in fd_{FD} and fd_{MD}.

At calculating the fractal dimension the results demonstrated that for the slope methods the fractal dimension is not just based on the count. The count for these methods can differ from what we would expect by a proportional relationship between box sizes. Compared to the fractal dimension fd_{LR}, the fractal dimension for the slope methods fd_{FD} and fd_{MD} would be higher or lower depending whether the count exceeds the proportional count or not. For future research, I suggest only to use the local fractal dimension fd_{LR}.

Global Fractal Dimension. The estimated global fractal dimension calculated with the slope of a regression is fd_{GS} = 1.33. Even though the determination coefficient was with 0.999 very high, the estimated fractal dimension differs from the real Koch Curve estimated fractal dimension 1.26 by 0.07. Similar to the estimated fractal dimension with the local methods, the estimated fractal dimension with the slope of a regression line focuses on differences between scales rather than the complexity of the pattern itself. For this reason, I suggest only to use the $fd_{GM(LR)}$ that is the mean of all fd_{LR}. For the Koch Curve this is $fd_{GS(LR)}$ = 1.28 and only 0.02 away from the true Koch Curve dimension (fd = 1.26).

How big has the difference in the fd to be to matter? It clearly depends on the context of application and even more on the consequence of a decision based on the estimated fractal dimension. An example where small differences can matter is medicine. The fractal dimension of organs, such as the kidney, could be estimated with a scan and three-dimensional Box-Counting. If a person goes to an annual medical check, and the estimated fractal dimension of their kidney decreases, this means that the function of the kidney to filter waste decreases, too. The consequence of the decision if the observed change in the estimated fractal dimension is normal or not, has an enormous consequence for the person between diagnosed as being healthy or seriously ill. Whether a small difference in this case matters, should be based on research with healthy and unhealthy people. An example where small differences appear to matter is provided by Reif, Qin and An[84] who obtained images of the microvasculature from a mouse ear at three consecutive days. They found a change in the fd of 0.003 and 0.004 between these days. They claim that this change is related to physical processes. The image was in rectangular form of the size ~ 3.5 × ~ 5.5 mm. With the Koch Curve, I showed that translating and rotating rectangles can cause a range of the estimated local fractal dimension of 0.23. This is much larger than the range Reif et. al based their decision on. Also, the Box-Counting method resulted in a estimated global fractal dimension of 1.33. There is a difference of 0.07 to the real fractal dimension 1.26 for the Koch Curve. This difference is much higher than the result from Reif et al.

On the other hand, even a big difference could be irrelevant. For example, managers at a clothes factory are confronted with the decision to choose between two fabrics with large differences in their estimated fractal dimension. They agree that the fabric with higher estimated fractal dimension looks more attractive, but they have to consider higher production costs, resulting in a higher retail price, arguably less sales volume and thus smaller turnover. From the perspective of the managers the outlook of higher profit might not be outweighed by

the preference of the more beautiful fabric. If the conclusion based on the estimated fractal dimension is critical, it may be wise to be more conservative and treating small differences in the estimated fractal dimension seriously. If there are more important variables, such as the profit, a difference in the estimated fractal dimension can be unimportant.

Chapter 5

Box-Counting Case Study

The aim of this case study is to analyse the fault system in New Zealand with the objective of identifying a spatial fractal pattern, that means to estimate the fractal dimension. Knowing the pattern improves the understanding of the underlying tectonic processes and may be used to identify undetected faults. For example, as a very simplistic illustration, if we consider $m = 10$, there are 10^2 boxes. If the estimated fractal dimension is 1.5, ≈ 32 out of the 100 boxes will be expected to be occupied. If the mapped fault lines only occupy 20 boxes, ≈ 12 of the empty boxes will be expected to have fault lines that are not mapped. Each of these 80 empty boxes (10^2-20) has the same probability to have unknown faults. The probability for each box can be increased and decreased by considering another magnification factor. At $m = 5$ we might find 11 occupied boxes, and this number matches the expected number. Therefore, empty boxes at $m = 5$ are not expected to have an unknown fault. The expectation is to find the estimated ≈ 12 boxes at $m = 10$ in one of the occupied areas at $m = 5$. Future earthquakes could be triggered at such an unknown fault line, and this is the reason why this research has the potential to support the prediction of earthquakes and to identify safe places for buildings.

The most important characteristics of a fractal object, its self-similarity and non-integer dimension, are useful for describing complex patterns (see Chapter 3. Mathematically generated artificial fractal objects such as the Koch Curve can show exact self-similarity meaning the fractal dimension is constant. In contrast, most objects in the real world have variation in the fractal dimension, and a more statistical approach, where variation is implicit, can be used. The Koch Curve is constructed by replacing exactly three line segment parts with four in each iteration. Nature does not work in such an exact algorithm. Fractals in nature are impacted by random events, for example a dry summer can influence the growth of a fractal tree, or, with the change of direction in the movement of the tectonic plates, the stress on areas is changing and influences the space of a rupture. Research in analysing complex real world objects, such as a fault system, in terms of fractal theory are motivated by the possibility of modelling the complexity and reflecting possible changes in the complexity on different scales.

The distributions of fault features, such as displacement and length, have been used to understand the evolution of fault systems. The motivation for this case study is based on research such as Lei and Kusunove[57], who investigated the fractal structure in the distributions of earthquake epicentres, active faults and rivers in Japan with Box-Counting (more detail in Section 2.7. They suggest that all three geological systems have a band-limited fractal structure shown in Figure 2.14. This success supported the decision to analyse the fault system on the South Island of New Zealand with the same approach. Lei and Kusunove[57] have not varied the properties in the user choices, e.g. they only used square boxes, whereas in this study both square and rectangular boxes are used.

The results in Chapter 4 showed that the estimated fractal dimension is influenced by user choices at magnification factor, interpolation method and length, shape of the box, and position of vector data on the grid as well as different methods for the estimation of the fractal dimension. The fractal dimesion for the Koch Curve has a known value of $fd = 1.26$, to which the estimated fractal dimension could be compared to. Since the fractal dimension value for the fault system is unknown, the analysis is based on the same approach as for the Koch Curve. The

following questions will be addressed. To what extent is fractal dimension influenced by the interpolation method and length, magnification factor, shape of the box, position of vector data on the grid? How do the different types of the estimated fractal dimension differ?

5.1 Data Preparation

For this case study I analysed the fault lines of the South Island of New Zealand. GNS Science supplied the dataset for this rectangular area as part of the geological map data *Quarter Million Mapping Seamless GIS 2012 pre-release version (QMAP)* This is New Zealand's 1:250 000 digital geological mapping project. These QMAP data are incomplete and are provided on an *as is, where is* basis and they may contain errors[(41)]. The dataset is explained in more detail in Section 2.12.

Covering the irregular shape of the South Island of New Zealand with a regular shaped box would include sea area for which no fault lines were available. This lack of data for the sea area would have distorted the analysis, and therefore I followed Walsh and Watteterson's (1993)[(106)] step of only analysing a regular proportion of the original area. I sought advice from two geologists and got the recommendation to analyse the area of the polygon with the points

$$
\begin{aligned}
&p_1\ (2.099\times10^6, 5.618\times10^6) \quad ,\\
&p_2\ (2.157\times10^6, 5.539\times10^6) \quad ,\\
&p_3\ (2.509\times10^6, 5.796\times10^6) \quad \text{and}\\
&p_4\ (2.451\times10^6, 5.875\times10^6)
\end{aligned}
$$

shown in Figure 5.1. This area is appropriate as it is south east to the South Alpine fault on the Pacific Plate and it also only covers inland area. The length is 435 km and the width is 97 km. The mapped fault lines in this area have a total length of 11,758 km. 96×324 km^2 of the total area were used for the analysis of four 96×96 km^2 areas. The QMAP was received as shapefile (.shp) that is a file format for storing geometric location and attribute information of geographic features. In the GIS language there are features and coordinates. Feature is a representation of a real-world object on a map - here, these are the QMAP fault lines and they are specified as *polylines features*. Coordinates (x,y) are the spatial information about a feature and define the location of fault lines.

The spatial information, i.e. the coordinates, of the features from the shapefile were read with the Matlab function `shaperead` and saved as a Matlab `mapstruct`, which is a geographic data structure (Appendix D.1). This geographic data structure has one element per each feature, in this application these are 6,503 elements for 6,503 fault lines in the study area. Each element has fields with spatial coordinates. The organization of the geographic data structure is called *element-by-element*, meaning the coordinates for each fault line are in separate fields as illustrated in Figure 5.2. This organization supports simple access to the complete information for one particular line.

In contrast to the *element-by-element* organization, a *plane* organization allows to easily access the entire coordinate values for all fault lines. I explain the concept of a *plane* organization with points described by only one pair of (X, Y) coordinates instead of polylines with more than two pairs of coordinate values. The *element-by-element* organization has 6,503 elements for 6,503 points. Each element has one field with the X coordinate value of one point. In contrast, the *plane* organization has only one field containing all 6,503 X coordinate values for all points illustrated in Figure 5.2. This simplifies the access of coordinates values of all features.

I changed the *element-by-element* organization into a *plane* organization with the function in Appendix D.1. The result are two fields, one for all 6,503 X and one for all 6,503 Y coordinates. In contrast to points, polylines are described with two or more pairs of (X, Y) coordinates. The 6,503 polylines in this case study are described

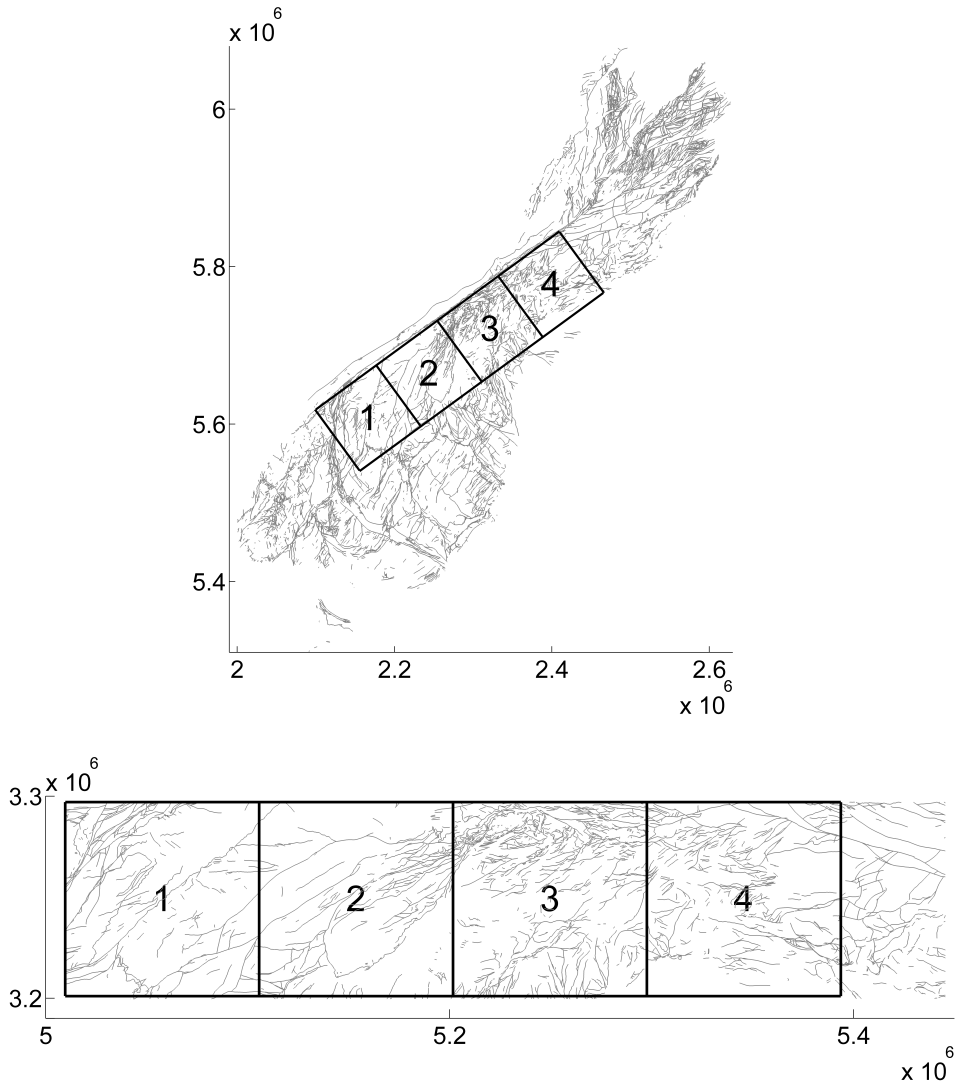


Figure 5.1 ***Area of analysis of the fault system in the South Island of New Zealand.*** *The shape of the Island is easy to recognize only by looking at its fault lines.*
top: *For this case study, the area of analysis is 324 km long and 96 km wide and has an direction of 36.16 degree NE. For exploring the effects the complete area was split into smaller regions, here four squares each of the size of* $96 \times 96\ km^2$.
bottom: *For easier extracting of square areas the original pattern is rotated so that there is no inclination.*

with 74,207 coordinate values before any interpolation. To separate the coordinate values of different polylines, there is an element `NaN` meaning *Not a Number* between the end of one polyline and the start of the next polyline. This is important, as otherwise Matlab would interpret all coordinates as only one polyline. I use the term *vectors* for the two fields with the (X, Y) coordinates.

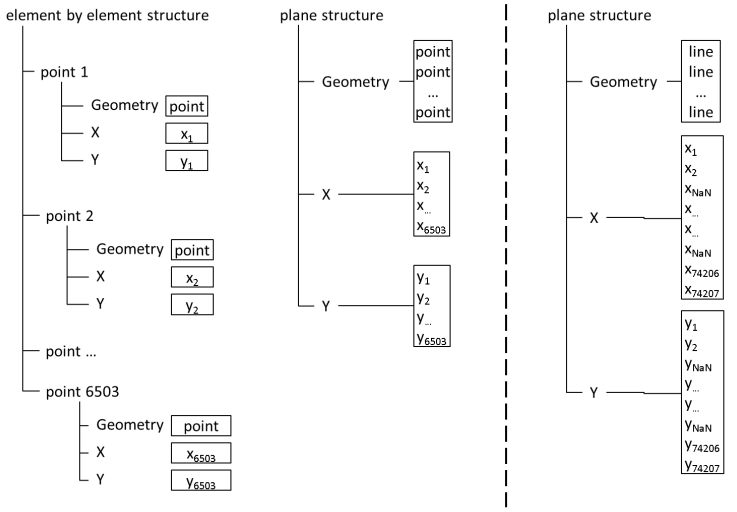


Figure 5.2 ***Element-by-element organization and plane organization of geographic data structures.*** *There are two ways to store attributes in a geographic data structure.*
left: *The element-by-element structure for points stores the coordinate values for each point separately.*
middle: *The plane structure for points stores all 6,503 coordinate values of all 6,503 points in one field.*
right: *The plane structure for polylines stores 74,207 coordinate values of all 6,503 polylines in one field.*

5.2 Method

I compared the effects at fault lines framed with the four squares illustrated in Figure 5.1. Each of the four squares has the size 96×96 km^2. I used the (X, Y) coordinates from the Matlab *plane* organization structure constructed earlier. The polygon that frames the dataset has the angle 36.16°north-east. The Matlab functions are provided in Appendix D.2 - D.10.

Data Rotation I rotated the complete dataset with the two X and Y vectors with

$$\begin{aligned} X_r &= X \times cos(-36.16°) - Y \times sin(-36.16°) \\ Y_r &= X \times sin(-36.16°) + Y \times cos(-36.16°) \quad , \end{aligned}$$

where X_r and Y_r are the rotated vectors (see Appendix D.2). The result is a rectangular area without inclination illustrated in Figure 5.1. The rectangular area parallel along the axes simplifies extracting fault lines in form of squares and rectangles.

Data Extraction The extracting is explained for the second 96×96 km^2 area shown in Figure 5.1 and Appendix D.4. At first the four boundaries of the rotated dataset were determined with the Matlab function `max()`

and `min()`

$$\begin{aligned} min(X_r) &= 5.010 \times 10^6 \quad \text{and} \quad max(X_r) = 5.445 \times 10^6 \\ min(Y_r) &= 3.200 \times 10^6 \quad \text{and} \quad max(Y_r) = 3.298 \times 10^6 \quad , \end{aligned}$$

where X_r and Y_r are the rotated vectors. This is the basis for determining the boundaries in the west X_w, east X_e, north Y_n and south Y_s of the second area with

$$\begin{aligned} X_w &= min(X_r) + h \\ &= 5.010 \times 10^6 + 96 \times 10^3 = 5.106 \times 10^6 \\ \\ X_e &= X_w + c \\ &= 5.106 \times 10^6 + 96 \times 10^3 = 5.202 \times 10^6 \\ \\ Y_n &= max(Y_r) - v \\ &= 3.297 \times 10^6 - 0 = 3.297 \times 10^6 \\ \\ Y_s &= Y_n - c \\ &= 3.297 \times 10^6 - 96 \times 10^3 = 3.201 \times 10^6 \quad , \end{aligned}$$

where h is the horizontal distance of the west border of the second square to the left boundary of the rotated dataset, c is the length of the square area and v is the vertical distance of the north border of the second square to the upper boundary of the rotated dataset. For the four 96×96 km^2 areas Y_n is always $max(Y_r$.

Polylines that are either completely or partly inside these borders are of interest. Polylines that are only partly inside are trimmed at the borders using the Matlab function `truncateAtBoundary`. The trimming is illustrated with a line segment with the start point p_s and end point p_e specified by Cartesian Coordinates in the vectors X_r and Y_r and the east border X_e of the second square illustrated in Figure 5.3.

The points of the line segment are

$$\begin{aligned} p_s &\quad (5.150 \times 10^6, 3.210 \times 10^6) \\ p_e &\quad (5.250 \times 10^6, 3.290 \times 10^6) \quad . \end{aligned}$$

At first, I identified the line segments crossing the east border defined by

$$b_p = \begin{cases} +1, & \text{if } X_p > X_e, \\ -1, & \text{otherwise,} \end{cases}$$

where positive b_p indicates that the point is right of the border and negative b_p that the point is left of the border. In the example,

$$\begin{aligned} X_{p_s} < X_e \quad &\Longrightarrow \quad b_{p_s} = -1 \\ X_{p_e} > X_e \quad &\Longrightarrow \quad b_{p_e} = +1 \quad . \end{aligned}$$

The difference between the assigned values b_{p_s} and b_{p_e} is *2* indicating the line segment is crossing the east border

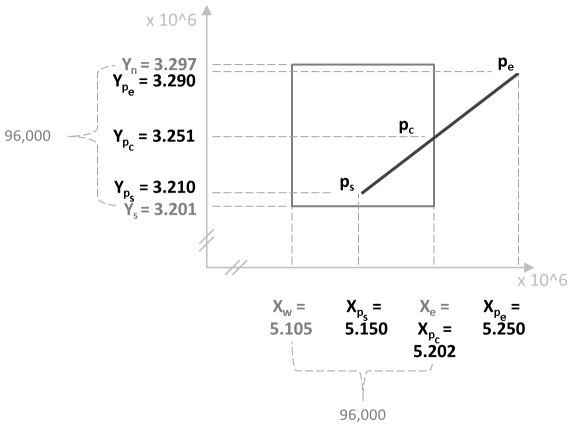


Figure 5.3 ***Extracting fault line data of a square area from the data of the total area.*** *Extracting line segments requires determining of the boundaries in the west X_w, east X_e, north Y_n and south Y_s. Then, line segments crossing a border are identified; here it is the east border of the second 96 $\times$96 km^2 area. The coordinate values for the crossing point p_c at the line segment that crosses the border are calculated. Finally, the point right at the east border, here the ending point p_e is replaced by the crossing point p_c.*

X_e. The difference between assigned b values for line segments not crossing the border would be *0*. This approach is used for all four borders.

After the line segments crossing the border had been identified, they were trimmed at the border. The x-coordinate is equal to the east border, and the y-coordinate can be calculated by reducing the lines' length with the scaling factor k

$$k = \frac{X_e - min(X_{p_s}, X_{p_e})}{|Y_{p_s} - Y_{p_e}|} \quad ,$$

where the nominator is the distance along the x-axis between the point left from the border to the east border X_e, and the denominator is the distance along the x-axis between the line segment's points. For the example this is

$$k = \frac{5.202 \times 10^6 - 5.150 \times 10^6}{|5.250 \times 10^6 - 5.150 \times 10^6|} = \frac{0.052 \times 10^6}{0.100 \times 10^6} = 0.052 \times 10^3$$

This scaling factor k is used for calculating the coordinate value Y_c of the crossing point p_c with

$$\begin{aligned} Y_c &= min(Y_{p_s}, X_{p_e}) + (|Y_{p_s} - Y_{p_e}| \times k) \\ &= min(3.210 \times 10^6, 3.290 \times 10^6) + (|3.210 \times 10^6 - 3.290 \times 10^6| \times 0.052 \times 10^3) = 3.251 \times 10^6 \quad . \end{aligned}$$

The crossing point p_c has the coordinate values

$$\begin{aligned} X_c = X_e &= 5.202 \times 10^6 \quad \text{and} \\ Y_c &= 3.251 \times 10^6 \quad . \end{aligned}$$

The end point p_e that was outside the square area is replaced with the crossing point p_c. This procedure is applied to all segments of the 6,503 polylines. All points that are still right to the east border X_e belong to line segments that do not cross the border and can be deleted. The resulting vector of the trimmed coordinates are called X_t and Y_t.

5.2.1 Converting Vector to Raster Data

Box-Counting is based on rasters. Occupied boxes in the raster reflect the polylines. The polylines are in vector form and have to be converted into raster form. I use the term *grid* for the initial pattern of connected empty box and the term *raster* after attributing *1* for *occupied* and *0* for *unoccupied* to each box. Resulting raster differ due to different interpolation length, magnification factor, shape of box and position of the vector data on the grid. These differences influence the count of occupied boxes and the estimated fractal dimension.

Interpolation. The influence of different interpolation methods and lengths on the estimated fractal dimension were explored. Interpolation restricts distances between two connected points of the fault lines to be equal or smaller than the chosen distance. The applied methods `doInterpm` and `doPythagoras` are explained in detail in Section 4.1 and Appendix D.9.

The interpolation length in units l_U was adjusted to the box edge length of the grid. The grid is constructed based on the magnification factor m and the box edge length is calculated by $\frac{96{,}000}{m}$. I calculated l_U with

$$l_U = \frac{96{,}000}{m} \times l_R \quad ,$$

where l_R is the relative interpolation length. The five different l_R I used are 0.3, 0.5, 0.7, 0.9, 0.99.

For example, for the combination of l_R = 03, m = 9,600 and a 96×96 km^2 area, the interpolation length l_U was calculated by

$$l_U = \frac{96{,}000}{m} \times l_R$$
$$l_U = \frac{96{,}000}{9{,}600} \times 0.3 = 3.$$

For the Matlab built-in function `doInterpm` this means that distances between two points along the x and y axes can not be longer than 3 m. For my modified function `doPythagoras` the interpolation length of 3m means that the line segment length between two points can not be longer than 3 m. `doInterpm` adds additional points evenly spaced; in contrast `doPythagoras` adds the points consecutively each after l_U.

These vectors were superimposed on the empty grids constructed with 24 different magnification factors. There were 11 rasters for each magnification factor: one for the original vectors, five for `doInterpm` combined with each interpolation length, and five for `doPythagoras` combined with each interpolation length. In total, there are 11 × 24 equal to 264 rasters. For each of these 264 combinations, the number of points describing the fault lines in the vector format, and the count of occupied boxes N describing the fault lines in raster format, and the estimated local fractal dimension fd_{LR} was determined. This process was applied to all four 96×96 km^2 areas.

Magnification Factor. Different magnification factors m influence the estimated fractal dimension fd_{LR}. The magnification factor m determines the size of boxes in a grid. With the increase of m the box size decreases.
The effect of 33 different magnification factors on the number of occupied boxes and the estimated fractal dimension were evaluated. The magnification factors range from 5 to 96,000 listed in Table C.3. The biggest magnification factor was 96,000 and connected to a box edge length of 1 metre. The smallest magnification factor was 5 and connected to a box edge length of 19,200 metre. The fault lines in vector format were interpolated with `doPythagoras` and a relative interpolation length l_R = 0.99. Section 4.1, and Appendix A.2 explained the creation of a raster with a Matlab `MapRasterReference` in detail (see Appendix D.8 and D.10).

For each of the 33 raster, the count of occupied boxes N, and the estimated local fractal dimension fd_{LR} was determined. This process was applied to all four 96×96 km^2 areas.

Shape of Boxes

So far, only square boxes had been used in this case study. Here, the influence of square and rectangular boxes on the fractal dimension fd_{LR} was compared. To clarify, rectangular boxes for a square area such as the first square of the size 96×96 km^2 in Figure 5.1 are actually squares. This is the reason for combining the four 96×96 km^2 areas to a single area of 96×324 km^2 for the illustration of different effects of square and rectangular boxes on the count of occupied boxes and the estimated fractal dimension.

The rasters were constructed with either square or rectangular boxes for 33 magnification factors ranging from 5 to 96,000. The superimposed fault lines in vector format were interpolated with `doPythagoras` and the relative interpolation length l_R = 0.99. For each of these 66 (=2×33) rasters the count of occupied boxes N and the estimated local fractal dimension fd_{LR} was determined.

Position of Vector Data on the Grid

To explore the influence of the location and orientation of the vector data on the grid on the count and the estimated fractal dimension fd_{LR}, the position of X_t and Y_t for the four squares of the size 96×96 km^2 on the grid was varied by translation and rotation. For each position one raster was created. The two possibilities to analyse the effect of position on the estimated fractal dimension is to translate and rotate either the position of the X_t and Y_t or the grid. For Box-Counting the relative position of line segments to each other is important rather than the absolute coordinates. This is why the vector data was translated and rotated, and the orientation of the grid was constant.

a) Translation. The points of the polylines were slid horizontally and vertically in 8 steps. The steps for the relative horizontal movement h_r and relative vertical movement v_r are 0.000, 0.125, 0.250, 0.375, 0.500, 0.625, 0.750, 0.0875 (see Appendix D.6. The value of v_r=0 indicates the original position. The absolute steps are calculated by

$$\begin{aligned} h_u &= h_r \times \frac{96,000}{m} \quad \text{and} \\ v_u &= v_r \times \frac{96,000}{m} \quad , \end{aligned}$$

where h_u and v_u are the horizontal and vertical steps in units.

The vectors for the fault lines X_t and Y_t had 64 different positions on the grid. The coordinate values for the line segments in these positions were calculated by

$$\begin{aligned} X_h &= X_t + h_u \\ Y_v &= Y_t + v_u \quad , \end{aligned}$$

where X_h and Y_v are the translated vectors.

The vectors X_h and Y_v, interpolated with `doPythagoras` combined with the relative interpolation length l_R=0.99, were superimposed on empty grids with square boxes for 12 magnification factors between 5 and 96.

It was important to keep the original left border X_w and bottom border Y_s for all rasters constant. Only the right border X_e and upper border Y_n were adjusted to the translated values. Without keeping X_w and Y_s constant, the final rasters would look the same as without any translation. In total, there were 768 (= 64 × 12) rasters. For each raster the count of occupied boxes N was evaluated, and the local fractal dimension fd_{LR} estimated. This process was applied to all four 96×96 km^2 areas.

b) Rotation of Square and Rectangular Areas. The vector data for the fault lines X_t and Y_t were rotated in steps of 5° from 0° to 90° with

$$\begin{aligned} X_d &= X\,t \times cos(d) - Y_t \times sin(d) \\ Y_d &= X\,t \times sin(d) + Y_t \times cos(d) \quad , \end{aligned}$$

where d is the angle in degree. The result was vectors for 19 different orientations.

The vectors X_d and Y_d, interpolated with `doPythagoras` and the relative interpolation length l_R=0.99, were superimposed on empty grids for 12 magnification factors between 5 and 96. For each of the 228 (= 19 × 12) newly constructed rasters, the count of occupied boxes N, and the estimated local fractal dimension fd_{LR} were evaluated.

Using square boxes, this process was applied to all four 96×96 km^2 areas, using rectangular boxes the combined 96×324 km^2 area was analyzed.

c) Rotation of Circular Areas. The results of the Koch Curve example in Chapter 4 showed the size of the raster changes with rotation. Using square boxes, this led to an increase of the number of raster rows and columns. Using rectangular boxes the effect of rotation was different box sizes for each orientation. An approach to keep both the number of rows and columns, and the box size constant, is using circular areas. Rotation of a circular area does neither change the raster size nor the box size.

A circular area with the radius 48 km was clipped at each of the four π×96× km^2 areas (see Appendix D.5). The further process of rotation and superimposing was the same as for the rotation of square areas.

5.2.2 Calculating Fractal Dimension

Different methods to calculate the fractal dimension can result in different estimated fractal dimensions. As explained in Section 4.2 all methods compare the relationship between the number of occupied boxes N and the magnification factor m on different scales, but they vary in the mathematical definition. Three local fractal dimensions fd_{FD}, fd_{MD} and fd_{LR}, and the two global fractal dimensions fd_{GM} and fd_{GS} were estimated. These five methods are explained in detail in Section 4.2.

The vectors X_l and Y_l interpolated with `doPythagoras` combined with the relative interpolation length l_R = 0.99 were superimposed on grids with square boxes for 33 magnification factors between 5 to 96,000. For each of the newly constructed 33 rasters the count of occupied boxes N, and the estimated local fractal dimension fd_{LR} were evaluated. The process was applied to all four 96×96 km^2 areas.

5.3 Results

Using the method with square boxes the analysis were applied to each of the four 96 × 96 km^2 areas shown in Figure 5.1. Using rectangular boxes and for comparing rectangular boxes with square boxes the complete 96×328 km^2 area was used. In the following section selected results are shown for each question of study.

5.3.1 Converting Vector to Raster Data

Interpolation. The number of points increases rapidly with interpolation. For the first 96 × 96 km^2 areas shown in Figure 5.1 the number of the original points is 13,761. Using the interpolation method `doInterpm`, the most extreme increase is from 13,761 to 674,918 points. This result occurs using magnification factor m = 9,600 and interpolation length l_R = 0.3 accented in bold in Table C.4. Using the interpolation method `doPythagoras` the increase of number of points to 759,020 at m = 9,600 and l_R = 0.3 is even more extreme than with `doInterpm`.

With smaller m less additional points were added. For $m \leq 16$ no points are added. This is because the shortest interpolation length for $m \leq 16$ is $\frac{96{,}000}{16 \times 0.3} = 1{,}800$ metre. This is shorter than the longest line segment with

1,740 metre for the first 96×96 km^2 area. I determined the length of the longest line segment by calculating the maximum length of all line segments using the Pythagorean theorem.

The variation in points affects the number of occupied boxes. For m = 9,600 the number of occupied boxes without any interpolation is 12,405. Using l_R=0.3 and `doInterpm` causes a jump up to N = 263,989 boxes, using `doPythagoras` it is even higher with N = 267,920 boxes. The range between the non-interpolated line segments and the interpolated line segments are 251,584 boxes for `doInterpm` and 255,515 boxes for `doPythagoras`. This range in counts transfers to diverse estimated fractal dimensions from fd_{LR} = 1.03 for the non-interpolated line segments and fd_{LR} = 1.36 for the interpolated lines.

With decreasing m, the range between fd_{LR} for different l_R decreases. At m = 9,600 the range in the fd_{LR} for all five relative interpolation lengths l_R using `doPythagoras` is 0.02. At m = 120 the range is 0 because for all five relative interpolation length the fractal dimension is fd_{LR} = 1.67. Here, the range of the occupied boxes between N = 2,961 and N = 3,008, has negligible influence on estimated fractal dimension (Figure 5.4).

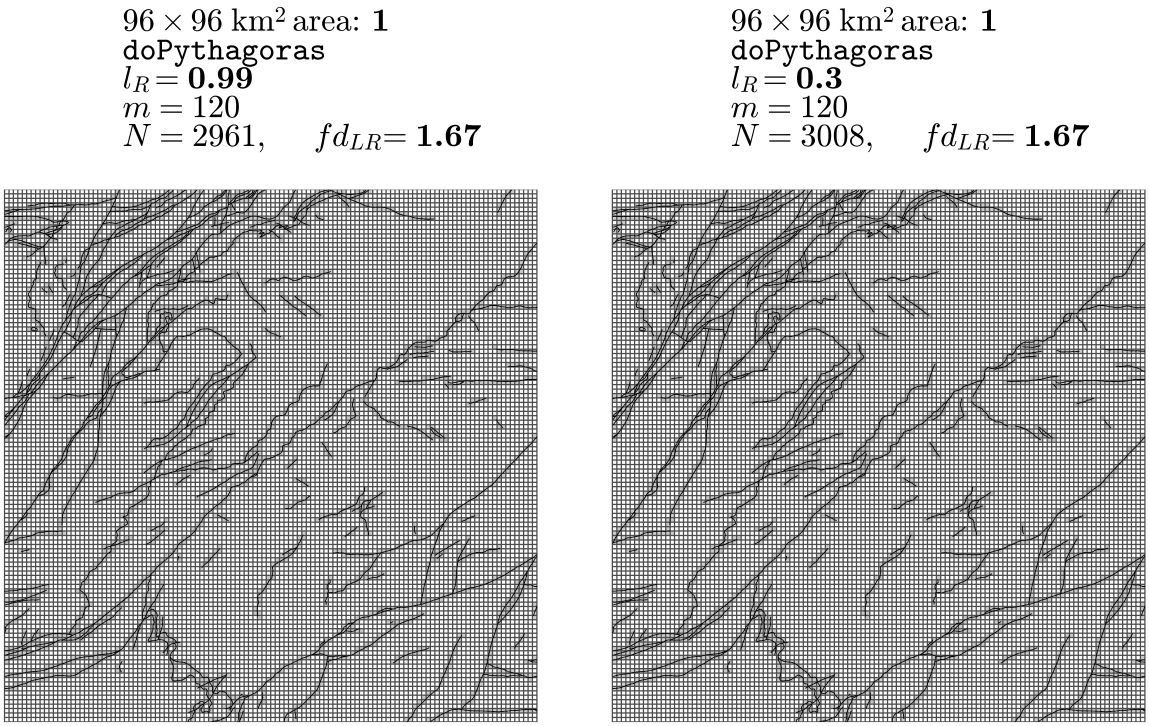


Figure 5.4 ***Influence of interpolation on the estimated fractal dimension fd_{LR}.*** *For the first 96 × 96 km^2 the relative interpolation lengths l_R barely influence the fd_{LR} at m = 120. The difference of N = 47 occupied boxes between l_R = 0.99 and l_R = 0.3 does not transfer to a range in the estimated fractal dimension.*
left: *N = 2,961 occupied boxes for l_R = 0.99 at m = 120 leading to fd_{LR} = 1.67.*
right: *N = 3,008 occupied boxes for l_R = 0.3 at m = 120 leading to fd_{LR} = 1.67.*

Magnification Factor. The higher the magnification factor the smaller the estimated local fractal dimension fd_{LR}. For the first 96×96 km^2 there is a fall from fd_{LR} = 2.00 to fd_{LR} = 1.40 with an increase of the magnification factor from m = 5 to m = 2,300 shown in Figure 5.5. At m = 5 all 25 boxes are occupied. The raster for the m = 2,300 is not suitable for visual demonstration, therefore the fall between fd_{LR} = 2.0 at m = 5 and fd_{LR} = 1.65 at m = 140 is illustrated in Figure 5.6. At m $\geq$ 48,000 the estimated fractal dimension might converge to fd_{LR} = 1.27 as illustrated in Figure 5.5 and Table C.3. Magnification was limited to m = 96,000 due to the restrictions in available computer processing capability.

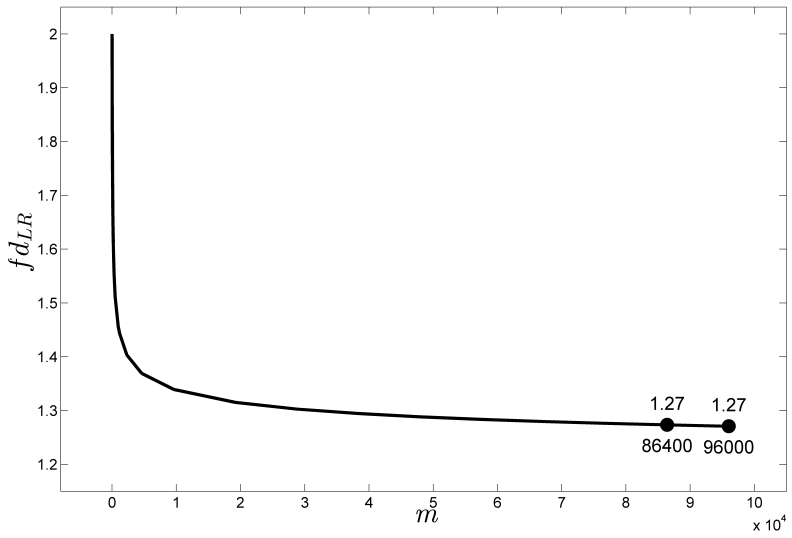


Figure 5.5 ***Influence of magnification factor m on fd_{LR}.*** *For the first 96×96 km^2 there is a decrease in the estimated fractal dimension from fd_{LR} = 2.00 to fd_{LR} = 1.40 with increase of the magnification factor from m = 5 to m = 2,300. For higher m the slope for the relationship between m and number of occupied boxes N levels out, meaning that the difference in the estimated fractal dimension gets smaller. At m = 96,000 the estimated fractal dimension is fd_{LR} = 1.27. The slope close to 0 indicates that convergence might be reached.*

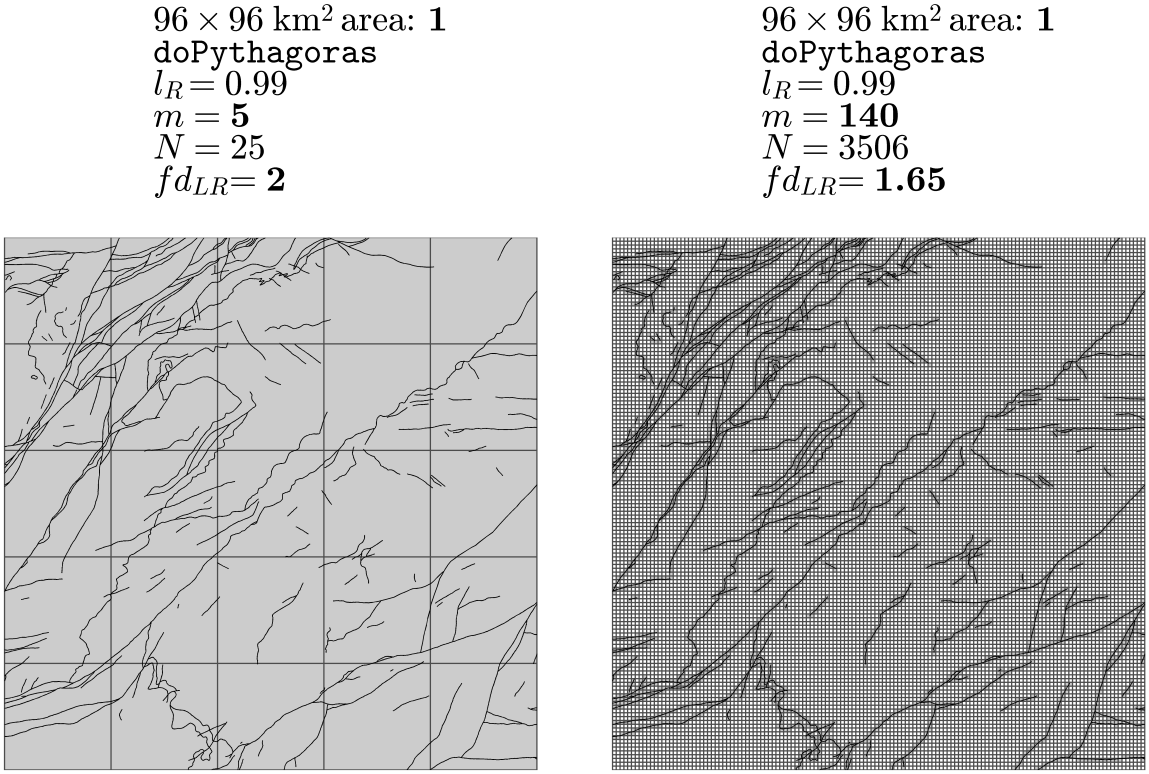
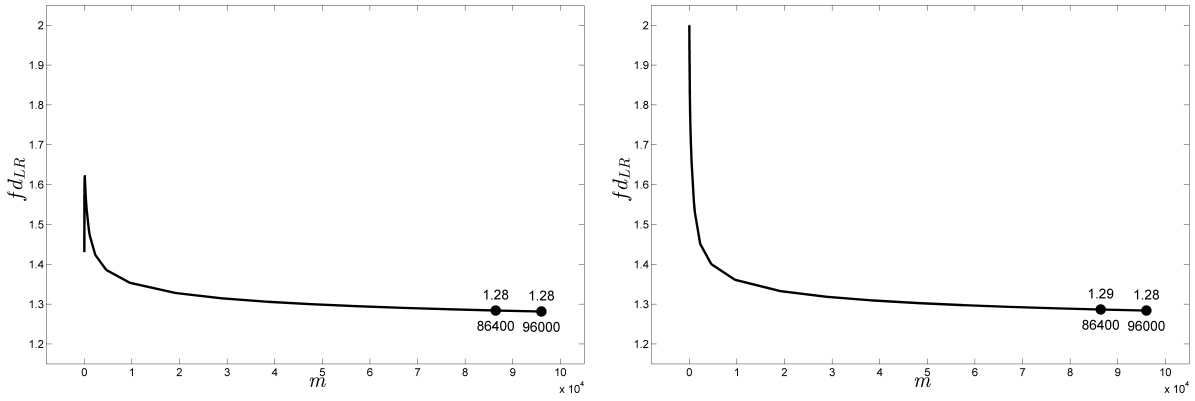


Figure 5.6 ***Influence of magnification factor m on the estimated fractal dimension fd_{LR} at m = 5 and m = 140.*** *For the first 96×96 km^2 area there is a steep decrease in the fd_{LR} of 0.35 between the m = 5 and m = 140.*
left: *25 occupied boxes for the relative interpolation length l_R = 0.99 at m = 5 leading to a fd_{LR} of 2.00.*
right: *3,508 occupied boxes for the relative interpolation length l_R = 0.99 at m = 140 leading to a fd_{LR} of 1.65.*

Shape of Boxes

For the combined 96×328 km^2 area, all rectangular boxes and all square boxes are occupied at m = 11 as illustrated in Figure 5.8 and Table C.5. However, the count of N = 121 rectangular boxes (11 rows × 11 columns) is higher than N = 33 square boxes (3 rows × 11 columns) due to the smaller size of rectangular boxes. This difference in the number of occupied boxes transfers to a higher estimated fractal dimension fd_{LR} = 2 using rectangular boxes than fd_{LR} = 1.46 using square boxes. The difference of 88 occupied boxes transfers to a difference of 0.54 in fd_{LR}.

With the increase of the magnification factor from m = 11 to m = 48 and m = 96, the range of occupied boxes between using square and rectangular boxes increases from N = 88 to N = 1,184 and N = 2,300, but the range in the estimated fractal dimension decreases from fd_{LR} = 0.54 to fd_{LR} = 0.30 and fd_{LR} = 0.22, respectively. This shows that with higher m the effect of the shape on fd_{LR} is smaller, even if the count N of square and rectangular boxes diverges. At m = 76,800, the estimated fractal dimension converge for both shapes to $fd_{LR}\approx$ 1.28 as illustrated in Figure 5.7.



Square boxes *Rectangular boxes*

Figure 5.7 ***Shape of boxes - square and rectangular boxes.*** *For the 96×324 km^2 area, the estimated fractal dimension fd_{LR} differs using square and rectangular boxes for magnification factor m ≤ 9,600. The biggest difference is 0.57 at m = 5. For both methods, the fd_{LR} converges at m = 96,000 to 1.28.*

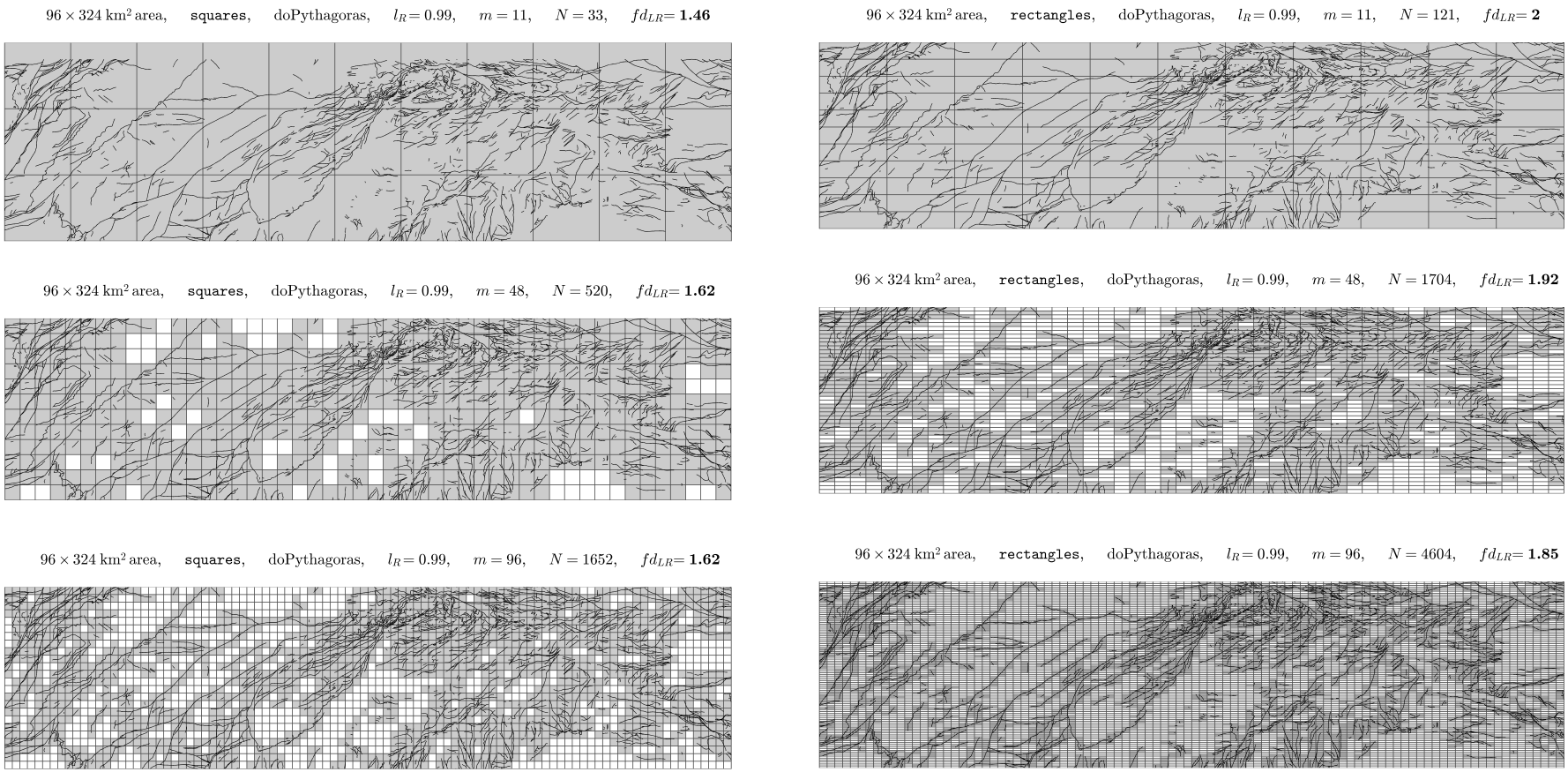


Figure 5.8 ***Shape of boxes - square and rectangular boxes at m = 11, m = 48 and m = 96.*** *The estimated fractal dimension fd_{LR} is always higher using rectangles than squares.*
top: *At the magnification factor m = 11 all square and rectangular boxes are occupied for the area 96 × 324 km^2.*
However, there is a difference in fd_{LR} of 0.54. The reason is that the square boxes are bigger. There are only N = 33 square boxes but N = 121 rectangular boxes.
middle: *With the increase of m the difference between the size of square and rectangular boxes gets smaller. This decrease in the size difference leads to a decrease in the range of N and fd_{LR}. The range in fd_{LR} decreases from 0.54 at m = 11 to 0.30 at m = 48.*

bottom: *The higher the magnification factor the smaller the range in the fd_{LR}. But still, at m = 96 there is a difference of 0.23 in fd_{LR}.*

Position of Vector Data on the Grid.

a) Translation. The translation of vector data on the grid influences the size of the raster. For translation, the east border X_e and north border Y_n are adjusted to the maximum of horizontally and vertically translated coordinate values. The west border X_w and south border Y_s remain the same, as otherwise all rasters would be identical. For building the empty grid with the adjusted borders, either the box size or the number boxes has to be increased. Here, the number of boxes was increased, but the box size $\frac{96{,}000}{m_o}$ was kept constant, where m_O was used as magnification factor m for the original position without any translation. However, calculating fd_{LR} with m_O for rasters whose number of boxes were increased can lead to illogical estimated fractal dimensions. One example for the fourth 96×96 km^2 square is

$$fd_{LR(m_{\mathrm{O}},\, h_r,\, v_r)} = \frac{log(N_{m_{\mathrm{a}},h,v})}{log(m_a)}$$
$$fd_{LR(5,\, 0.000,\, 0.250)} = \frac{log(29)}{log(5)} \qquad = 2.09 \quad ,$$

where h_r and v_r are the relative horizontal and vertical translation. The fd_{LR} = 2.09 is not correct calculated as the estimated fractal dimension has to be between topological and embedded dimension, in our case

$$\begin{array}{ccccc} td & \leq & fd & \leq & ed \\ 1 & \leq & fd & \leq & 2 \end{array} \quad ,$$

where *td* is the topological dimension and *ed* is the embedded dimension. For this reason, the original magnification factor m_O was adjusted to the higher number of rows or columns, called adjusted magnification factor m_A. All translation positions required only up to one more row and one more column of boxes in the raster, and therefore have the same m_A. As m_O for the original position is different from m_A for the translated position, only the translation positions were compared with each other.

For the fourth 96×96 km^2 square, the most extreme difference in fd_{LR} of different translation positions of the vector data on the grid is 0.19 (Table C.6). This difference is found between the positions

$$fd_{LR(m_{\mathrm{a}},\, h_{\mathrm{r}},\, v_{\mathrm{r}})} = \frac{log(N_{m_{\mathrm{a}},h,v})}{log(m_a)}$$
$$fd_{LR(6,\, 0.000,\, 0.875)} = \frac{log(25)}{log(6)} \qquad = 1.80 \quad \text{and}$$
$$fd_{LR(6,\, 0.125,\, 0.125)} = \frac{log(35)}{log(6)} \qquad = 1.98 \quad ,$$

where m_A is the adjusted magnification factor, h_r the relative horizontal translation and v_r the relative vertical translation. The difference of N = 10 boxes leads to a range of 0.18 in fd_{LR} at m_A = 6.

With the increase of the m_A the range in fd_{LR} is decreasing. At m_A = 12 the most extreme cases are

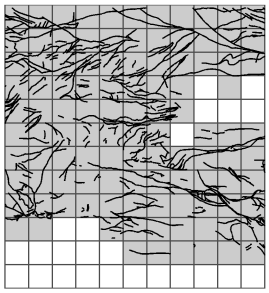
$$fd_{LR(12,\, 0.000,\, 0.875)} = \frac{log(108)}{log(12)} \quad = 1.88 \quad \text{and}$$
$$fd_{LR(12,\, 0.625,\, 0.250)} = \frac{log(128)}{log(12)} \quad = 1.95 \quad ,$$

illustrated in Figure 5.9. The difference of N = 20 boxes leads to a range of 0.07 in fd_{LR} at m_A = 12.

$h_R = \frac{0}{8}, v_R = \frac{7}{8}$

$e = \frac{96000}{11}, \quad m_a = 12$

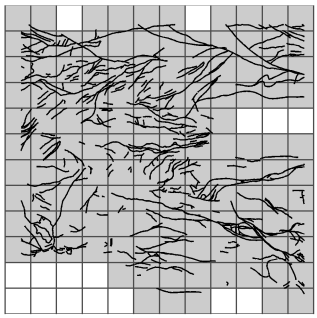
$N = 108, \quad fd_{LR} = \mathbf{1.88}$



$h_R = \frac{5}{8}, v_R = \frac{2}{8}$

$e = \frac{96000}{11}, \quad m_a = 12$

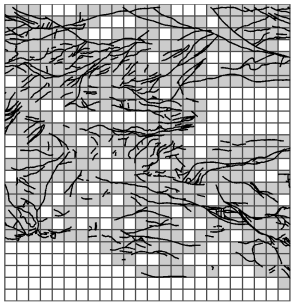
$N = 128, \quad fd_{LR} = \mathbf{1.95}$



$h_R = \frac{0}{8}, \quad v_R = \frac{6}{8}$

$e = \frac{96000}{24}, \quad m_a = 25$

$N = 416, \quad fd_{LR} = \mathbf{1.87}$



$h_R = \frac{1}{8}, \quad v_R = \frac{2}{8}$

$e = \frac{96000}{24}, \quad m_a = 25$

$N = 452, \quad fd_{LR} = \mathbf{1.9}$

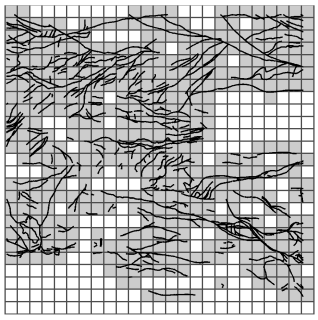


Figure 5.9 ***Translation using square boxes for the fourth 96×96 km² area.***
top: *At the adjusted magnification factor m_A = 12 the minimum number of occupied boxes is N = 108 for an relative upward movement of h_R = 0.875 of the box edge length. The maximum is N = 128 for h_R = 0.625 of the box edge length for a vertical translation of v_R = 0.250 of the box edge length. The range of 20 boxes leads to a range in the fd_{LR} of 0.07 from $fd_{LR_{12,\ 0.000,\ 0.875}}$=1.88 to $fd_{LR_{12,\ 0.625,\ 0.250}}$=1.95.*
bottom: *At m_A=25 the minimum N = 416 is for h_R = 0.750. The maximum N = 452 is for h_R = 0.125 and for v_R = 0.250. The range of 36 boxes leads to a range in the fd_{LR} of 0.03 from $fd_{LR_{25,\ 0.000,\ 0.750}}$=1.88 to $fd_{LR_{25,\ 0.125,\ 0.250}}$=1.90.*

At m_A = 25 the difference is even smaller with the extreme cases

$$fd_{LR(25,\ 0.000,\ 0.750)} = \frac{log(416)}{log(12)} = 1.87 \quad \text{and}$$
$$fd_{LR(25,\ 0.125,\ 0.250)} = \frac{log(452)}{log(12)} = 1.90 \quad ,$$

illustrated in Figure 5.9. The range of N = 36 boxes at m_A = 25 is higher than N = 20 boxes at m_A = 12. However, due to the higher magnification factor the higher range in N leads to a smaller range in fd_{LR}.

This pattern of increasing ranges in N and decreasing ranges in fd_{LR} continues with the increase of m_A. At m_A = 96 the estimated fractal dimension of all translation positions converges to $fd_{LR} \approx 1.72$. This shows that a high enough m can counter the effect of the position.

b) Rotation of Square and Rectangular Areas. Similar to the translation, an adjustment of m_O to m_A is necessary for rotation. With the rotation, the distance between two X and two Y coordinate values for a line segment is changing along the x- and y- axes. With this change, also the size of the raster is changing illustrated for the angle 0° and 40° in Figure 5.10. The magnification factor was adjusted to the new raster size, and the box size was kept constant as it was done for translation. While m_A was the same for all translated positions, m_A for rotated positions is different. The highest m_A occurs at the rotation with 40°, 45°and 50°. Because of different m_A, the box edge length e is used as reference.

Using square boxes, for the third 96×96 km^2 area the most extreme difference in fd_{LR} of different rotation positions of the vector data is found at $e = \frac{96{,}000}{10}$. The difference of 0.20 in fd_{LR} is between the positions

$$fd_{LR(m,\ d)} = \frac{log(N_{m_A,\ d})}{log(m_a)}$$
$$fd_{LR(14,\ 40)} = \frac{log(114)}{log(14)} = 1.79 \quad \text{and}$$
$$fd_{LR(10,\ 0)} = \frac{log(98)}{log(10)} = 1.99 \quad ,$$

where d is the angle in degree, and m either m_O for the original position with d = 0 or m_A for the rotated position with d = 40.

At the box edge length $e = \frac{96{,}000}{24}$ the difference in fd_{LR} is 0.17 and found between

$$fd_{LR(34,\ 40)} = \frac{log(515)}{log(34)} = 1.77 \quad \text{and}$$
$$fd_{LR(24,\ 0)} = \frac{log(483)}{log(24)} = 1.94 \quad ,$$

$N_{(14,\ 40)}$ = 114 and $N_{(34,\ 40)}$ = 515 at 40° is more than $N_{(10,\ 0)}$ = 98 and $N_{(24,\ 0)}$ = 483 at 0°. However, fd_{LR} at 40° is smaller than at 0° because of the higher m_A in the denominator.

In contrast to using square boxes, adjusting the magnification factor for rectangular boxes is dispensable as this method always adjust the size of the boxes but keeps the magnification factor m_O. Therefore m_O can be used as reference to compare the different positions of the 96 × 384 km^2 areas.

At almost all m_O the maximum number of occupied boxes was more than twice the minimum number of occupied boxes. For example, at m_O = 24 the number of occupied boxes is N = 212 at 50° and N = 524 at

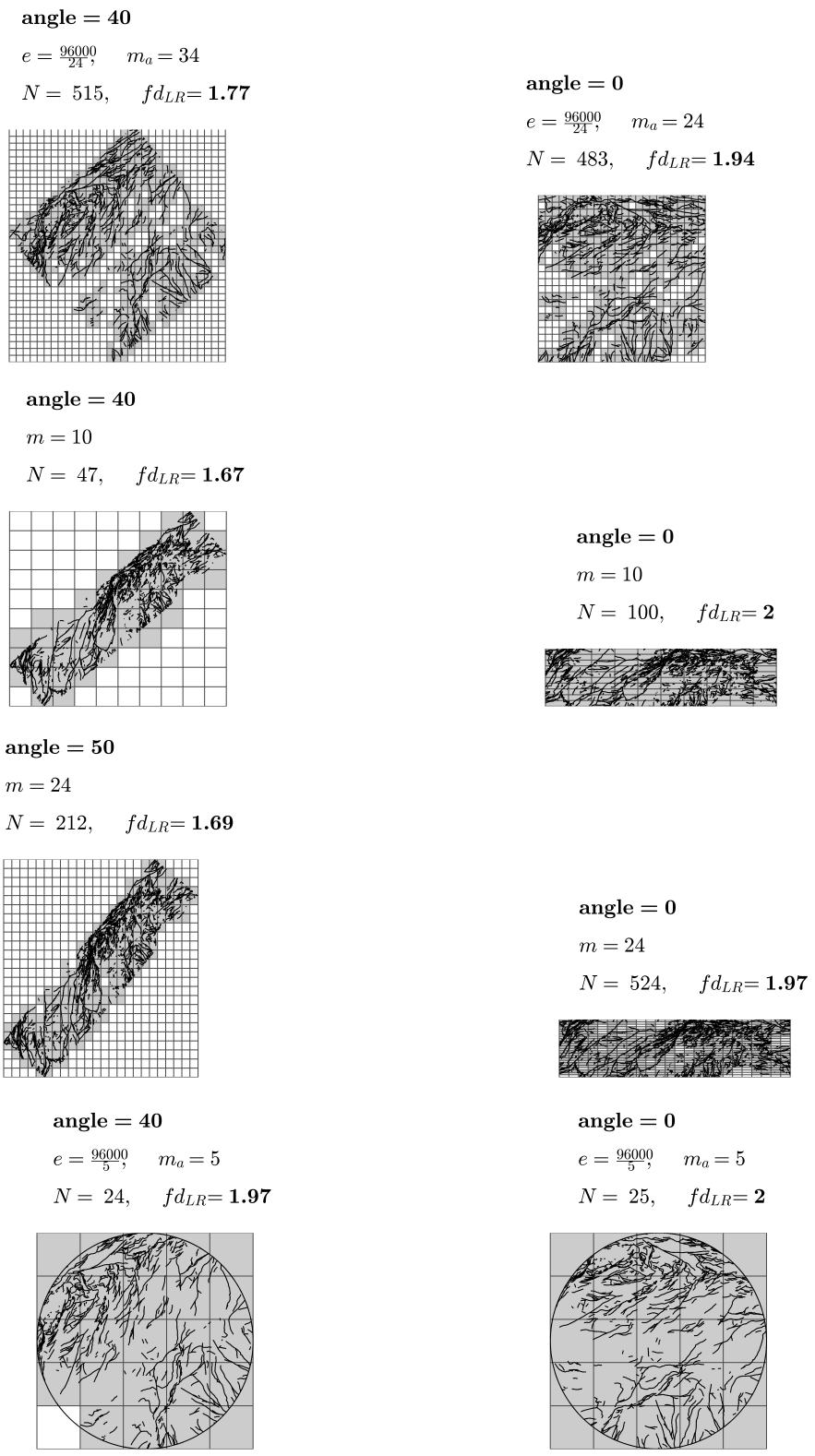


Figure 5.10 ***Rotation of vectors on grid using square and rectangular boxes for square, rectangular or circular areas.***
top: *Rotation using square boxes for the third 96×96 km² area at box edge length e = $\frac{96,000}{24}$. The rotation of the vectors leads to a change in the size of the raster. The estimated fractal dimenion $fd_{LR,\ 24,\ 0}$ = 1.94 is higher than $fd_{LR,\ 34,\ 40}$ = 1.77, even though the number of occupied boxes $N_{24,\ 0}$ = 483 is lower than $N_{34,\ 40}$ = 515.*
upper middle: *Rotation using rectangular boxes for 96×324 km² area at magnification factor m = 10. The $fd_{LR,\ 10,\ 0}$ = 2.00 is 0.33 higher than $fd_{LR,\ 10,\ 40}$ = 1.67. However, the boxes at angle 40° are much bigger.*
lower bottom: *Rotation using rectangular boxes for 96×324 km² area at m = 24. The range in the log ratio local fd is 0.28 at m = 24 for different rotation positions of the 96×384 km² area.*
The $fd_{LR,\ 24,\ 0}$ = 1.97 is 0.28 bigger than $fd_{LR,\ 24,\ 50}$ = 1.69.
bottom: *Rotation using square boxes on circular area for the third 96×96 km² area at m = 5 or e = $\frac{96,000}{5}$. The range in the fd_{LR} for circular areas is contrary to the result for the complete square area. The most extreme difference in the fd_{LR} is 0.02 at box edge length e = $\frac{96000}{5}$. This is much smaller than the difference of 0.17 for the complete square area.*

0° shown in Figure 5.10. This is an increase of 147 percent. The range in fd_{LR} is 0.28 between

$$fd_{LR(24,\ 50)} = \frac{log(212)}{log(24)} \quad = 1.69 \quad \text{and}$$
$$fd_{LR(24,\ 0)} = \frac{log(524)}{log(24)} \quad = 1.97 \quad .$$

At m_O = 10, at 0°all 100 rectangular boxes are occupied, at 40°only 47 illustrated in Figure 5.10. Note, the box size at 40°is much bigger. The range of 53 boxes transfers to a range in fd_{LR} of 0.33 between

$$fd_{LR(10,\ 40)} = \frac{log(47)}{log(10)} \quad = 1.67 \quad \text{and}$$
$$fd_{LR(10,\ 0)} = \frac{log(100)}{log(10)} \quad = 2.00 \quad .$$

Note, the size of the rectangular boxes for this 96 × 384 km^2 area is changing with each rotation. Taking this into account, it cannot be confirmed that the rotation alone influence the range in fd_{LR}.

c) Rotation of Circular Areas. For circular areas, the fractal dimension fd_{LR} is underestimated, as the number of fault lines for the clipped $pi \times \frac{96}{2} \times \frac{96}{2}$ km^2 area is less than the number of fault lines in the complete 96×96 km^2 square area. However, here, the range in the fractal dimension is more critical than the fractal dimension value itself. The result of the range for the circular areas contradicts the result for the complete square area. For the third 96×96 km^2 area the most extreme difference in fd_{LR} is 0.02 at box edge length = $\frac{96{,}000}{5}$ as illustrated in Figure 5.10. This is much smaller than the difference of 0.20 in fd_{LR} for the complete square area. The range in the estimated fractal dimension decreases with the decrease of the box edge length.

5.3.2 Estimating Fractal Dimension

The estimated fractal dimension for the third 96×96 km^2 area in Figure 5.1 shows that with the increase of m all three estimated local fractal dimension tend to decrease as illustrated in Figure 5.11. Whereas the fd_{LR} continuously decreases, the fd_{FD} and fd_{MD} change direction. It is not that both fd_{FD} and fd_{MD} fluctuate, they are often < 1.0. This means they are between the dimension of a point with dimension td = 0 and a straight line with dimension td = 1. This result is not expected looking at the third square in Figure 5.1. This frequent change in the direction of fd_{FD} and fd_{MD} was discussed in Section 4.3. Here, fd_{FD} and fd_{MD} are always smaller than fd_{LR}. This difference transfers to the estimated global fractal dimensions , where $fd_{GM(FD)}$ = 1.24 and $fd_{GM(MD)}$ = 1.25 are smaller than $fd_{(GM)LR}$ = 1.64. The estimated global fractal dimension calculated with the slope fd_{GS} = 1.10. As discussed in Section 4.3 the same fd_{GS} could be estimated for more or less complex pattern.

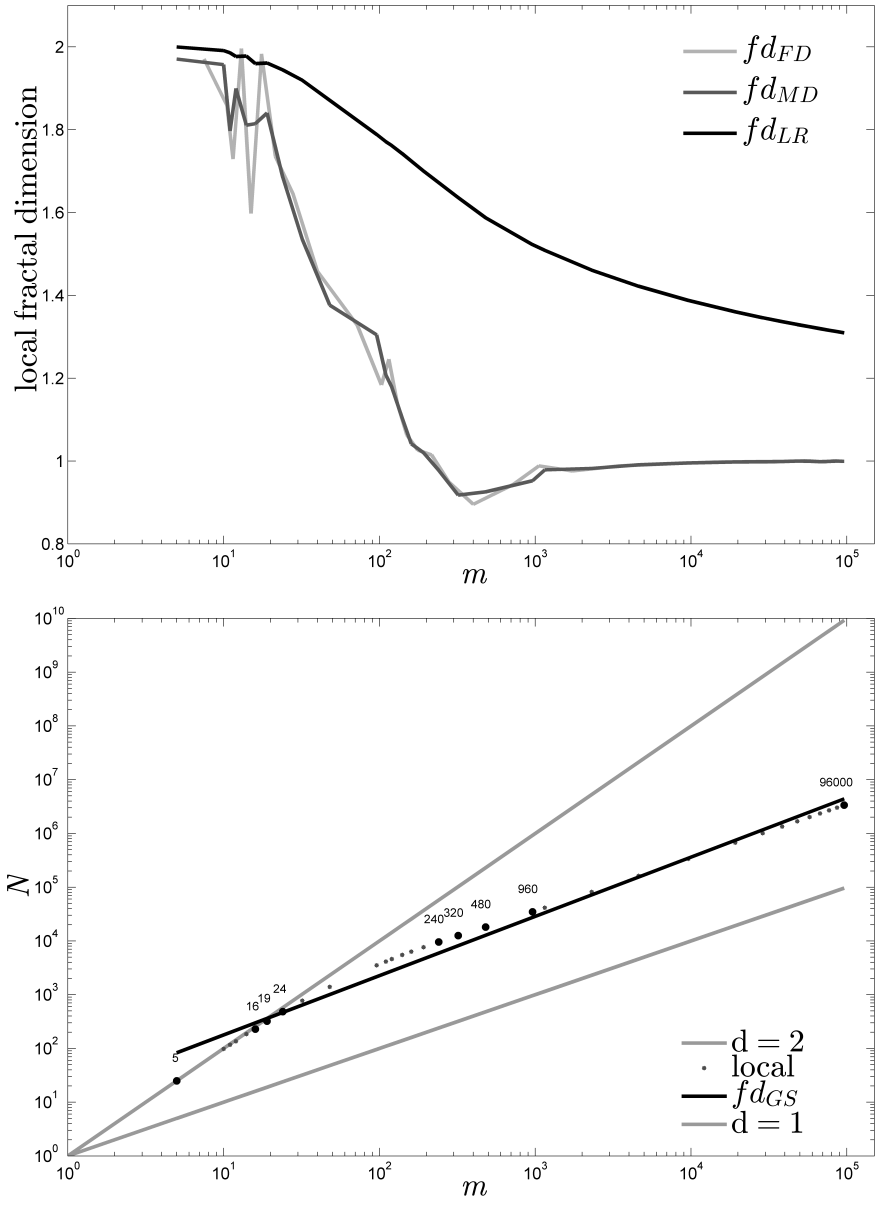


Figure 5.11 ***Estimated local and global fractal dimension for the third 96×96 km² area.***
top: *Whereas the estimated fractal dimension $fd_{LR(m)}$ decreases with increase of the magnification factor m, both fd_{FD} and fd_{MD} increase and decrease for different values of m.*
bottom: *The steepness of the regression line indicates the estimated global fractal dimension fd_{GS}. The line d = 1 and d = 2 represent dimension 1 and 2, respectively. The numbers above the points are magnification factors.*

5.4 Summary and Discussion

This case study has highlighted potential differences in the estimated fractal dimension because of the shape of the pattern and user choices. I summarized the most extreme cases for the fault lines in Table C.11.

Interpolation. For the first 96 $\times$ 96 km^2 area there are differences in the estimated fractal dimension due to the different interpolation methods and lengths. With increase of the interpolation length, the estimated fractal dimension decreases. The reason is that the number of additional points decreases, and thus the number of occupied boxes. Both interpolation method result in very similar estimated fractal dimension. The advantage of using `doPythagoras` is to know that each box occupying a line segment equal or longer than the interpolation method is identified as occupied.

Magnification Factor. With the increase of the magnification factor the estimated fractal dimension fd_{LR} decreases. For the first 96 $\times$ 96 km^2 there is a range of 0.73 in the estimated fractal dimension. This clearly shows, that the scale is important when applying Box-Counting to fault lines. Differences in the estimated fractal dimensions at low magnification factors are greater than differences at high magnification factors. The estimated fractal dimension appears to converge at 1.27 at the magnification factor 96,000. This result supports that Box-Counting is appropriate for large magnification factors.

Shape of Boxes. Using square or rectangular boxes at the 96 $\times$ 324 km^2 area leads to a difference up to 0.54 in fd_{LR}. With the increase of the magnification factor the difference in the size of rectangular and square boxes decreases and this transfers to a smaller range in the estimated fractal dimension. For magnification factor 19,200 and higher, the estimated fractal dimension for both shapes is 1.33. The estimated fractal dimension appears to converge at 1.27 at the magnification factor 96,000. This result suggest that the shape of the box is important for low magnification factors rather than high magnification factors.

Position of Vector Data on Grid. Another factor that affects the number of occupied boxes is the positioning of the vector data on the grid. The horizontal and vertical translation with squares for the fourth 96 $\times$ 96 km^2 area results in a range of 0.07 in the estimated fractal dimension at the adjusted magnification factor 12. The range for rotation is even bigger. The rotation using squares for the third 96 $\times$ 96 km^2 area results in a range of 0.20 in the estimated fractal dimension at the box edge length of $\frac{96{,}000}{10}$ metre. The rotation using rectangles leads to a maximum number of boxes that is often more than twice as many as the minimum number of occupied boxes. The reason is the change of the size of rectangular boxes during rotation. At magnification factor 10 there is a range in the estimated fractal dimension of 0.33 between the angle 0° and angle 40°.

For the translation and rotation, I suggest not to target the maximum or minimum estimated fractal dimension but to minimize the space surrounding the pattern. In Figure 5.10 the original patterns are very closely surrounded, whereas the rotated patterns have more empty space around them. In fact, the estimated fractal dimension of a pattern could be decreased by just superimposing it on a bigger grid. This was done with the rotation of rectangles. Comparing the grid size the right one in Figure 5.10 is about three times as big as the left one.

For the rotation of square and rectangular boxes, the size of the raster changed, and either the number of boxes or the size of the boxes had to be adjusted. The size of the raster for the rotation of circular area was constant. There is almost no difference at all magnification factors in the estimated fractal dimension for different orientation of circular areas. This suggests, that observed differences for square and rectangular areas are due to change of the raster size rather than the pattern itself.

Estimating Fractal Dimension. Concerning the choice of calculation of the estimated local fractal dimension, the results confirm the suggestion in Section 4.3 only to use the use the log ratio between number of occupied

boxes and magnification factor, as both slope techniques focus on the relation between scales rather than on the complexity of the pattern. The same reason justifies not to use the global fractal dimension calculated as slope.

Choice of properties. Overall, the question of how to choose the properties in the fractal analysis is not easily answered. One major issue is the shape of the box. It depends what the user is expecting. If the user expects a fractal pattern such as the artificial fault lines in the Sierpinski carpet illustrated in Figure 5.12 (left), the use of square boxes is suggested. But the fault network also can be stretched and rectangular boxes may be more appropriate as illustrated in Figure 5.12 (right). In both cases, the estimated fractal dimension would be 1.77. However, the conclusions are different. Finding the estimated fractal dimension for the fault lines in rectangular boxes can help to understand the geophysical processes. The change from square to rectangular boxes can be achieved either by stretching or squeezing. For the fault lines it should be the pushing theory, as the Pacific and the Australian plate move towards each other.

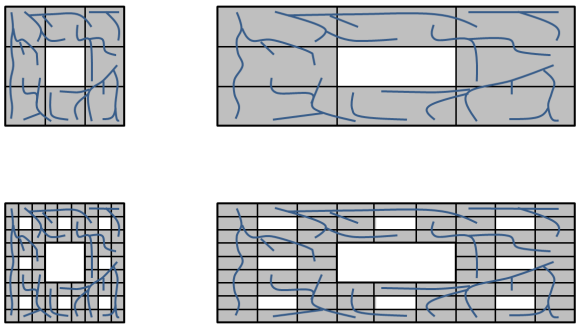


Figure 5.12 ***Same estimated fractal dimension for squares and rectangular boxes.*** *The stretching of the artificial fault network leads to rectangular boxes. For both shape of boxes the estimated fractal dimension is 1.77. In order to see the difference, it is important to communicate how the fd_{LR} was calculated.*
left: *The fd_{LR} is found using square boxes.*
right: *The fd_{LR} is found using rectangular boxes.*

Comparison of results for Box-Counting for the Koch Curve and fault lines. The estimated global fractal dimension calculated as mean of the local log ratio fractal dimensions is 1.28 for the Koch Curve. The estimated local log ratio fractal dimension is 1.27 at the highest magnification factor. Both values are similar to the true value of 1.26. Here, for the fault lines the mean of the estimated global fractal dimension calculated as mean of the local log ratio fractal dimensions is 1.64. The estimated local log ratio fractal dimension is 1.27 at the magnification factor 96,000. The difference of 0.37 for the fault lines is much higher than for the Koch Curve. The visual representation of the fault lines arguably appears to be more complex than the Koch Curve. This would recommend to use the mean of the log ratio local fractal dimensions, here 1.64. However, the range of 0.72 in the log ratio local fractal dimensions is very big, and using the mean appears not to be appropriate.

Many fault lines in the QMAP are approximate, inferred, or concealed rather than accurate. The choice of the magnification factor 96,000 referred to a box edge length of 1m. This scale could have been too small to describe inaccurate fault lines. Furthermore, the dataset is incomplete, and the counts of occupied boxes was probably lower than with a dataset that mapped all fault lines. Thus, the estimated local log ratio fractal dimension 1.27 describing the complexity of the fault lines is possibly underestimated. It is suggested to investigate how to determine an appropriate magnification factor.

Comparison with Japan. The decision to analyse the fault system on the South Island of New Zealand with Box-Counting was based on a similar study from Lei & Kusunose[57] in Japan. They found an estimated fractal dimension calculated as slope between $0.9 \leq fd \leq 1.1$ for box sizes < 13 km and $1.3 \leq fd \leq 1.5$ for box sizes > 13 km

as illustrated in Figure 2.14. The value of fd = 1.0 corresponds to the dimension of a straight line. As discussed in Section 4.3, the slope methods focus on the differences between values instead of focusing on the values themselves. This means that $fd \approx 1.0$ found in Japan could have been calculated for a different complex pattern as long as the values have the same relationship to each other, that means the height of the regression line moves upwards or downwards.

Given that the results in Japan could be confirmed with the fd_{LR}, then the estimated fractal dimension for fault lines in Japan and in New Zealand follows a similar pattern of decreasing fd_{LR} with increasing m, that is connected to decrease in box size. The break in the estimated fractal dimension is at the box edge length e = 13 km. This length equates to m = 7.38 for a 96×96 km^2 area. I compare results at $m \leq 7$ (equates to $e \geq$ 13,7 km) with the larger Japanese scale, and $m \geq 8$ (equates to $e \geq$ 12 km) with the smaller Japanese scale.

The estimated fractal dimensions in New Zealand are always higher than in Japan. For the smaller Japanese scale, the estimated fractal dimension in Japan between 0.9 and 1.1 is smaller than the minimum estimated fractal dimension of $fd_{LR(96000)}$ = 1.27 in the first 96×96 km^2 area in New Zealand. For the bigger Japanese scale, the estimated fractal dimension in Japan is between 1.3 and 1.5. In New Zealand the range for box sizes >13 km is between $fd_{LR(6,\ 0.000,\ 0.875)}$ = 1.8 in the fourth 96×96 km^2 area and $fd_{LR(5)}$ = 2 in the first 96×96 km^2 area. Higher estimated fractal dimensions in New Zealand indicate a geometrically more complex fault system than that in Japan.

However, these fractal dimensions are estimated and can differ from the true fractal dimension. Testing the null hypothesis that the difference between the estimated fractal dimension is zero can lead to more confidence. The testing can be done with a *paired sample t-test*. The estimated fractal dimensions of both New Zealand and Japan can be matched at each magnification factor m. It is important that the fractal dimension in both countries are estimated in the exact same way. I suggest to use the local dimension fd_{LR} because this focuses on the values N and m rather than on differences of these values.

After further testing, statistically significant differences between New Zealand and Japan could be explained by underlying differences in the pressure of the tectonic plates for both plates. Whereas the area of analysis for this case study has a transform boundary, the plate boundary in Japan is a convergent plate boundary indicated in purple and the subduction zones in blue in Figure5.13. Earthquakes at convergent plate boundaries are deeper than at transform plate boundaries. Deep earthquakes normally cause less damage on earth's surface, such as faults, than shallow earthquakes with the same magnitude. Japan's deeper earthquakes could be the reason that the fault system is less complex than in New Zealand.

This explanation cannot be supported with the knowledge of different types of rock and soil condition in both countries. Before an earthquake is triggered, the pressure or stress of the tectonic plates builds up strain on the rocks. It depends on the type of rock when the strain exceeds the strength of the rocks. Soft rocks rupture at a lower stress than hard rock. Transferring this theory to Japan and New Zealand, Japan with the less complex fault system should have harder rock than New Zealand. This hypothesis contradicts with the knowledge that at a convergent plate boundary, such as in Japan, the harder and therefore heavier oceanic rock dips below the lighter continental rock. This means that in Japan the lighter rock is on the top. The area in New Zealand used for this case study is on the Pacific Plate, an almost entirely oceanic plate with the heavier rock. In this theory, the upper rock in New Zealand should be heavier than the upper rock in Japan. The rock in Japan should rupture at a lower stress level and therefore have a more complex fault system.

Differences in Areas. It appears that the estimated fractal dimension is converging at $fd_{LR(96000)}$=1.27 in the first 96×96 km^2 area and at $fd_{LR(96000)}$=1.31 in the third 96×96 km^2 area. In future research a randomization test can be used to test whether the difference of 0.04 in the estimated fractal dimensions between Japan and New Zealand is statistically significant. The observed difference of 0.04 is compared with the distribution of differences in the fractal dimension obtained by randomly reordering the fault lines between Japan and New Zealand. The significant level of the difference in the estimated fractal dimension is the proportion of differences that are as

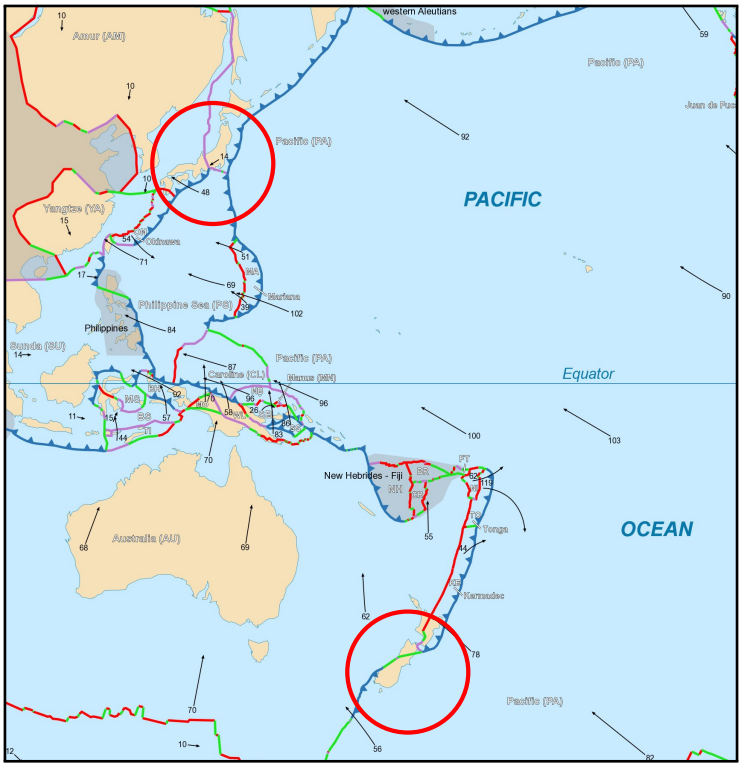


Figure 5.13 ***Tectonic Plates with their movement direction.*** *The South Alpine fault is a transform plate boundary, meaning the two plates are passing each other. (indicated in green in red circle). In contrast, the plate boundary in Japan is convergent, meaning the plates are moving toward each other (indicated in purple and blue in red circle).*
(from newworldencyclopedia)[73]

extreme or more extreme than the observed difference of 0.04. In Chapter 6 the process is presented in more detail for the application of randomization to distributions of the orientation of fault lines in different areas.

Chapter 6

Testing Fault Line Orientation

In this chapter I compare the geographic orientations of fault lines in different areas in New Zealand. New Zealand is located on the border of Australian and Pacific plates in the southern Pacific Ocean. Both the Australian and the Pacific plate move towards each other. As the direction of the movement of the two plates is at an angle rather than 180 degrees, the pressure between the two plates is not constant along the New Zealand landmass. The change in the pressure is visible at the change of the orientation of the Alpine fault shown in Figure 2.8. Knowing the pattern of the orientation of fault lines can improve the understanding of underlying geological processes. The orientation of the fault lines can be summarized in a rose plot shown in Figure 6.1. This graphic tool is an angle histogram and shows the distribution of angles grouped according to their degrees. Meteorologists use it to give a f2compact view of the wind direction distribution at one particular location. Unlike wind, fault lines do not have direction, they have orientation. This means that the start and end point of a fault line are exchangeable. A fault line with 45° pointing north-east can be treated as a fault line with 225° pointing south-west. In the following I suggest how to use rose plots, not just as a descriptive tool, but also to distinguish between a random and a statistically significant difference in the orientation of fault lines in different areas. A permutation Kolmogorov-Smirnov test was adapted to compare the orientation of multiple groups.

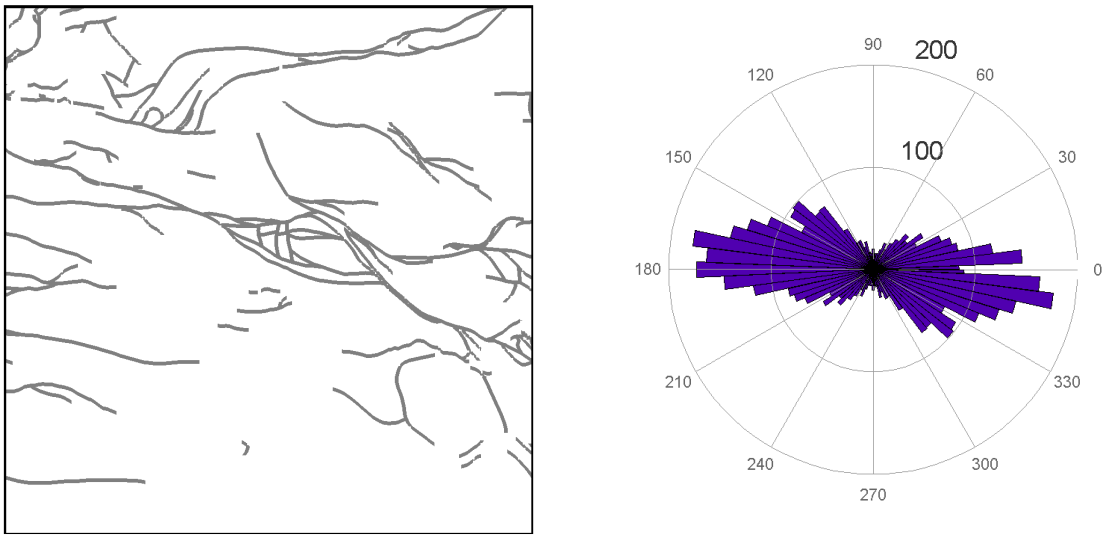


Figure 6.1 ***Roseplot describing the orientation of fault lines.*** *Rose plots are angular histograms showing the frequency and the orientation of the fault lines.*

6.1 Comparing Orientation of two areas

Fault lines have different orientations. To compare the orientation of fault lines in two areas, one approach is to calculate the angles of the fault lines, with respect to the origin, and compare the mean values. The mean of two fault lines with orientation 30° and 170° can be either 100°, 190°, 280°or 10°as illustrated in Figure 6.2. This makes the definition of average somewhat ambiguous and thus not useful. The ambiguity increases with the number of fault lines being studied, therefore, I consider distributions rather than central tendencies.

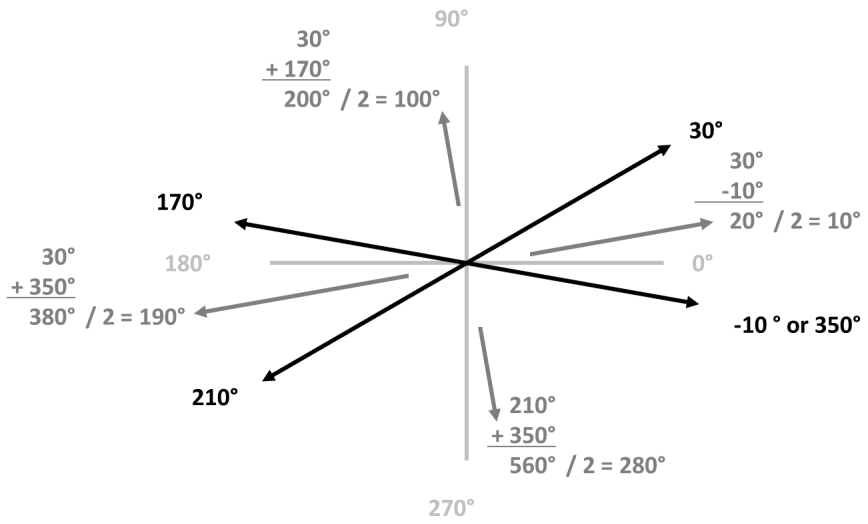


Figure 6.2 ***Calculation of the mean orientation.*** *The average orientation of two fault lines can be evaluated in four different ways. The mean of a fault with orientation 30° or 210° and a second fault line with -10° or 170° can be either 100°, 190°, 280°or 10°*

Kolmogorov-Smirnov test for two samples (2KS). The two-sample Kolmogorov-Smirnov (2KS)[92] test is used to decide if two groups have statistically significantly different distributions. I illustrate the process of the 2KS test with an artificial dataset of fault lines in two areas. The angles of the fault lines for each area are

area 1: 10 , 16 , 19 , 19 , 22 , 25 , 25 , 28 , 135 ,
area 2: 13 , 35 , 125 , 128 , 131 , 131 , 137 , 137 , 140 , 143 , 143 , 146 .

The orientations of the fault lines are illustrated in Figure 6.3. Note, that each angle is represented with its original value as well as mirrored, e.g. the angle 10° is shown twice, once as 10° and once with 190°. The angle spectrum from 0° to 180°is used for the calculations. The null hypothesis H_0 is that the distribution of the orientation is the same in the two areas.

The 2KS test compares the empirical cumulative distribution functions (ecdf) shown in Figure 6.4 (left). The ecdf F_n for n angles with degrees D is defined by

$$F(d) \quad = \quad \frac{1}{n} \quad I(D_i \leq d) \, ,$$

where $I(D_i \leq d)$ indicates the number of angles $\leq$ degree d. In other words, F_n is the proportion of angles $D \leq$

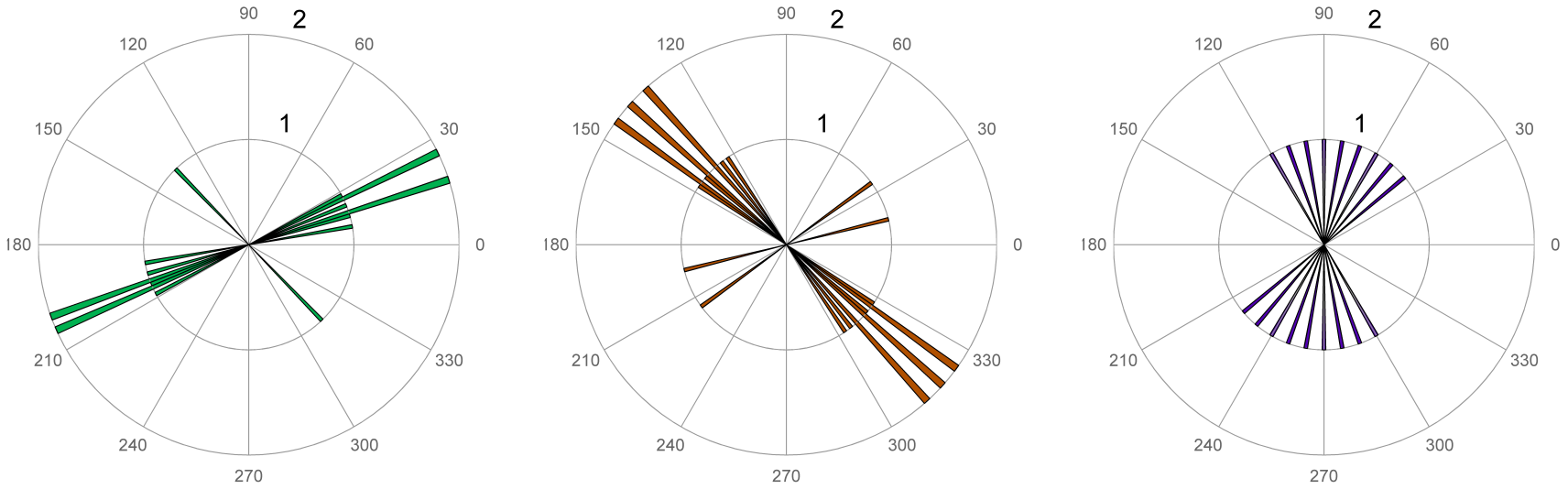


Figure 6.3 ***Rose plots representing the orientation of fault lines in three areas (artificial dataset).*** *Rose plots are angle histograms. Each fault lines segment is represented by its angle. The angles are grouped by their degrees. For the illustration each group is represented with at most two angles. The frequency is visualized by the circles.*
The first two groups are used for explaining the Kolmogorov-Smirnov test for two groups (2KS) and permutation.
All three groups form the basis for describing the Kolmogorov-Smirnov test for more than two groups (MKS).

degree d. The 2KS test statistic D_{2KS_r} is the maximum absolute vertical distance between the two ecdfs defined as

$$D_{2KS_r} = max \left| F_{(1)}(d) - F_{(2)}(d) \right| ,$$

where the maximum is over all angles d, and $F_{(1)}$ and $F_{(2)}$ are the ecdfs for areas 1 and 2. The maximum vertical distance between the two ecdfs is $0 \leq D_{2KS_r} \leq 1$. The higher the D_{2KS_r}, more likely it is that the angles are from two different distributions. In the example, the test statistic $D_{2KS_r} = 0.81$ exceeds the critical value $D_{2KS_c} = 0.55$ for the two groups with 9 and 12 angles and $\alpha = 0.05$ [72] (see Figure 6.4). Thus the null hypothesis is rejected and there is sufficient evidence that the two areas have different distributions of fault line orientation. For the comparison of samples with more than 12 elements the critical value D_{2KS_c} can be calculated as

$$D_{2KS_c} = c(\alpha)\sqrt{\frac{n_1 \times n_2}{n_1 + n_2}}$$

where n_1 and n_2 are sample sizes and $c(\alpha)$ is a coefficient published by Smirnov [92].

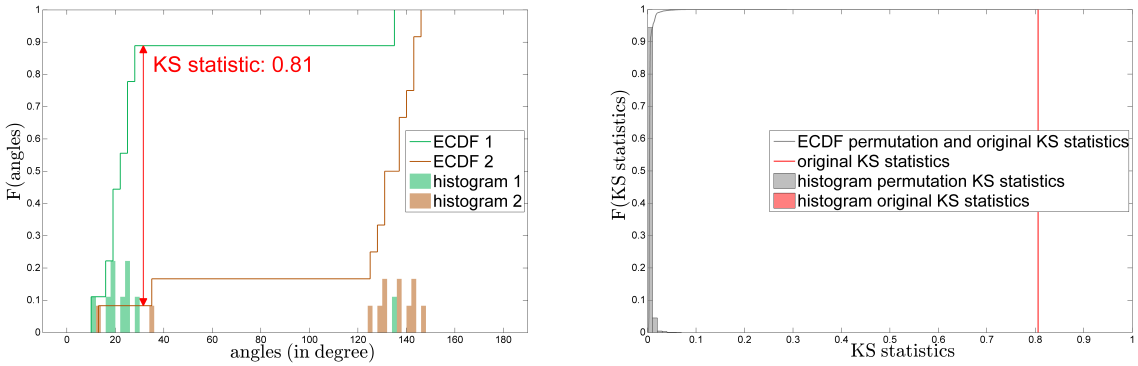


Figure 6.4 ***Classical and permutation 2KS for an artificial dataset of fault lines in two areas.***
left: Classical 2KS test. *Green and brown lines each correspond to an ecdf of the orientation of fault lines in different areas. The histograms represents the frequency of the angles of these fault lines. Histograms can be thought to be unwrapped roseplots. The red arrow indicates the test statistic $D_{2KS_r} = 0.81$ which is the maximum vertical difference between the two ecdfs. The test statistic $D_{2KS_r} = 0.81$ is greater than $D_{2KS_c} = 0.55$ at $p < 0.05$. H_0 is rejected and the result suggest that the two areas have different distributions.*
right: Permutation 2KS test. *This is the sampling distribution of all 9,999 D_{2KS_p} and the D_{2KS_r}. The bin for the $D_{2KS_r} = 0.81$ is too small too be visible. Almost all D_{2KS_p} are in the first bin, thus the ecdf increases steeply. The red line at $D_{2KS_r} = 0.81$ shows that no D_{2KS_p} is than D_{2KS_r}. H_0 is rejected and the result indicates that the difference between the two distributions is statistically significant.*

In this study, there were more than two groups. I compare the distributions for 18 areas of the size 48 km^2 shown in figure 6.5. Since the 2KS is only comparing two areas, I adjust the test to compare the distributions of multiple groups. The test statistic of the new *Kolmogorov-Smirnov test for multiple groups (MKS)* is the absolute maximum vertical distance between the lowest and the highest ecdf at each angle d between any two areas. At the current stage there are no critical values for the multiple KS test, and so the MKS is constructed as permutation test.

6.2 Permutation test

Permutation tests are based on the premise that if all areas have the same distribution, then assignment of fault lines to areas should not matter. Therefore the original assignments are permuted N_{sim} times and the test statistic is evaluated for each permuted dataset to produce the sampling distribution for the test statistic. The observed value

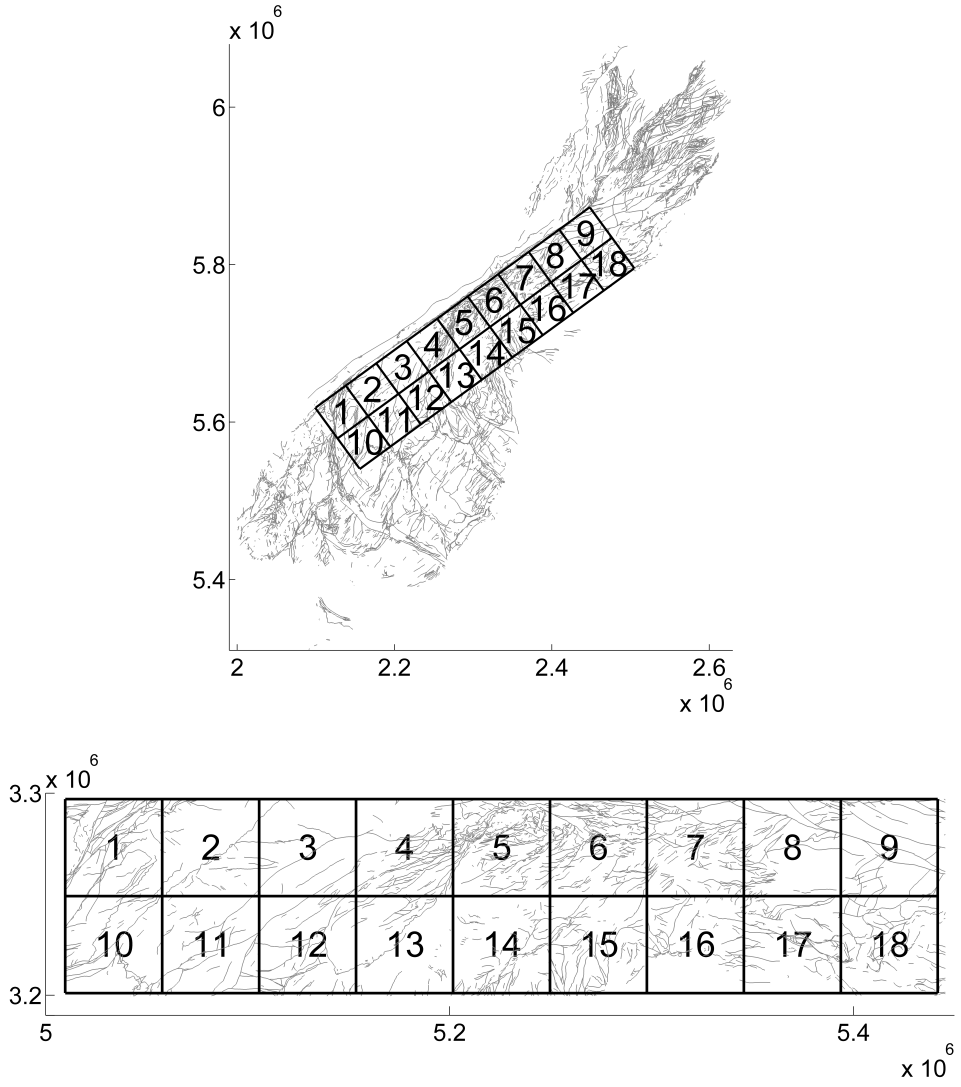


Figure 6.5 ***Area of analysis of the Fault System of the South Island of New Zealand.*** *For exploring the orientation of the fault lines the complete study area of 432 × 96 km^2 was split into smaller regions, here 18 squares each of the size of 48×96 km^2. For easier extracting of square areas the original pattern is rotated so that there is no inclination. Note, the roseplots in this case study show the orientation of the rotated angles.*

of the test statistic is then compared to that distribution to get the p-value and thus the idea of how extreme or unusual that test statistic value is under the null hypothesis. N_{sim} should be high enough to reach convergence.

I illustrate the permutation 2KS test using the same data set as for the classical 2KS test and N_{sim} = 9999. The sampling distribution of all D_{2KS_P} and the observed value D_{2KS_r} is in Figure 6.4. The p-value is calculated as

$$p \quad = \quad \frac{1}{n_{sim}} \quad I(D_{2KS_p} > D_{2KS_r}) \, ,$$

where $I(D_{2KS_p} > D_{2KS_r})$ indicates the number of permutation test statistics D_{2KS_p} that are greater than the observed value D_{2KS_r}. In the example, none of the 9,999 D_{2KS_p} is greater than D_{2KS_r} and therefore p < 0.0001. H_0 is rejected and there is sufficient evidence that the difference in the distributions of the orientation of fault in these areas is not random. This is the same result as for the classical 2KS test.

6.3 Kolmogorov-Smirnov test for multiple groups (MKS)

For comparing m groups one must carry out $\frac{m!}{2}$ pairwise 2KS tests. The type 1 error rate inflates with the number of tests and interpretation of the results is complicated. The adjusted Kolmogorov-Smirnov test for multiple groups (MKS) is a single test, which answers the question *"Are any two areas out of m different?"*. If a statistically significant difference is found, the 2KS can be used to find where it is exactly. The test statistic D_{MKS_r} is defined as

$$D_{MKS_r} \quad = \quad \max_{i,j} \quad (\quad | \quad F_i(d) \quad - \quad F_j(d) \quad | \quad) \, , \quad i, j = 1, 2, ..., m$$

for all possible pairs (i, j) over all angles d, where F_i and F_j are the ecdfs for the areas i and j. For an illustration, the three areas from Figure 6.3 with the angles

group 1: 10 , 16 , 19 , 19 , 22 , 25 , 25 , 28 , 135 ,
group 2: 13 , 35 , 125 , 128 , 131 , 131 , 137 , 137 , 140 , 143 , 143 , 146
group 3: 40 , 50 , 60 , 70 , 80 , 90 , 100 , 110 , 120 .

have D_{MKS_r} = 0.89. In the example, none of the D_{MKS_p} is greater than D_{MKS_r} and therefore p < 0.001 (see Figure 6.6). The null hypothesis is rejected and there is sufficient evidence that at least two out of three areas have statistically significantly different distributions.

6.4 Case Study

For this case study the fault lines of the South Island of New Zealand are analyzed. I use the same dataset as in Chapter 5. It is described in Section 2.12 and 5.1. The Matlab functions are provided in Appendix D.11 and D.12).

Figure 6.5 shows the 18 areas of the size 48 km^2 whose distributions are compared. The orientation angle θ of a fault line defined by Cartesian Coordinates x and y is

$$\theta = tan^{-1} \quad \frac{y}{x}$$

and the fault line length l is

$$l \quad = \quad \sqrt{x^2 + y^2} \, .$$

The MKS is used to test whether there are any different pattern between the 18 areas, and if a statistically significant difference is found the 2KS is used to identify between which of the eighteen areas the difference is.

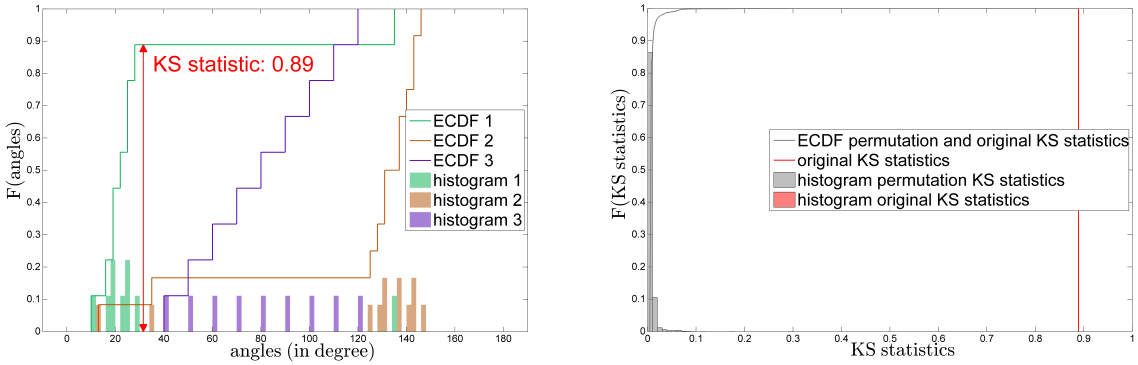


Figure 6.6 ***MKS for an artificial dataset of fault lines in three areas.***
left: *Green, brown and purple lines each correspond to an ecdf of the orientation of fault lines in different areas. The histogram represents the frequency of the angles of these fault lines. Histograms can be thought to be unwrapped roseplots. The red arrow is the maximum vertical difference between the three ecdf and used as test statistic D_{MKS_r}. Here, $D_{MKS_r} = 0.89$.*
right: *The gray line corresponds to an ecdf of the sampling distribution from the permutation process including the 9,999D_{MKS_p} for the permutations and $D_{MKS_r} = 0.89$. The red line for the $D_{MKS_r} = 0.89$ shows that no D_{MKS_p} is greater than D_{MKS_r}. H_0 is rejected and the result indicates that any two of the three distributions are statistically significantly different.*

There are two versions, the first is with unweighted orientation and the second with orientation weighted by the length of the fault line. The second procedure is as follows: The length of each fault lines is converted to a multiple of 100 m and this multiple is the weight. For example, a line segment with length 456 m and angle 30° is rounded up to 500 m and has a weight of 5. There were 999 permutations for the assignment of fault lines to different areas (see Appendix D.11 and D.12.

6.5 Results

For the purpose of clarity, I only show the result for three of the eighteen 48×48 km^2 areas illustrated in Figure 6.5. The number of angles are 1,967 in the 11th area, 1,200 angles in the 13th area and 1,894 angles in the 17th area.

6.5.1 Orientation of Unweighted Fault Lines

The orientation of the unweighted fault lines for the 11th, 13th and 17th area are shown in roseplots in Figure 6.7. The roseplots show that the purple area 17 has more fault lines than the green area 11 and brown area 13. The number of angles are 1,967 in the 11th area, 1,200 angles in the 13th area and 1,894 angles in the 17th area.

Kolmogorov-Smirnov test for multiple samples (MKS) The observed difference of $D_{MKS_{r(11,13,17)}} = 0.44$ appears to be larger than what would be expected by chance if the three areas had the same distribution of fault line orientations. The sampling distribution of the 999 $D_{MKS_{p(11,13,17)}}$ and $D_{MKS_{r(11,13,17)}}$ in Figure 6.8 clearly shows that there is no $D_{MKS_{p(11,13,17)}}$ greater than $D_{MKS_{r(11,13,17)}}$. There is no variation in the p-value, thus even more permutation are expected to lead to $p < 0.001$. This supports that the difference $D_{MKS_{r(11,13,17)}}$ is not random and that at least two out of three distributions are statistically significantly different.

Kolmogorov-Smirnov classical and permutation test for two samples (2KS) With the 2KS test the numerical result from the MKS is specified that suggested that there is a difference in the distribution of angles in any two of the three areas. The test statistic $D_{2KS_{r(11,17)}} = 0.44$ is the highest found between the ecdfs of area 11 and 17, $D_{2KS_{r(13,17)}} = 0.42$ for area 13 and 17 is very similar. The $D_{2KS_{r(11,13)}} = 0.04$ for area 11 and 13 is the lowest illustrated in Figure 6.9.

The classical 2KS indicates that $D_{2KS_{r(11,17)}} = 0.44$ and $D_{2KS_{r(13,17)}} = 0.42$ are statistically significantly different from 0 ($p < 0.001$). For these cases the H_0 is rejected and the conclusion is that there is a statistically significant difference in the distribution of fault line orientations between these areas. The difference between area 11 and 13 with $D_{2KS_{r(11,13)}} = 0.04$ is not statistically significant different from 0 ($p > 0.08$).

The permutation 2KS confirms the classical test results. After 999 permutations none of the D_{2KS_p} are higher than the $D_{2KS_{r(11,17)}} = 0.44$ and $D_{2KS_{r(13,17)}} = 0.42$. There are some D_{2KS_p} greater than $D_{2KS_{r(11,13)}} = 0.04$, thus the difference is statistically not significant ($p > 0.07$) (see figure 6.10).

6.5.2 Orientation of Weighted Fault Lines

The orientation of the weighted fault lines for the 11th, 13th and 17th area are shown in roseplots in Figure 6.11. The roseplots with the weighted fault lines look more balanced than the roseplots with the unweighted fault lines. This indicates that the total length of fault lines are similar in all three areas.

Kolmogorov-Smirnov test for multiple samples (MKS). The observed difference of $D_{MKS_{r(11,13,17)}} = 0.5$ for the weighted angles is even greater than $D_{MKS_{r(11,13,17)}} = 0.44$ for the unweighted angles. The sampling distribution for the weighted angles highlights that none of the D_{MKS_p} is greater $D_{MKS_{r(11,13,17)}} = 0.5$, thus leading to $p < 0.001$ (Figure 6.12). This supports that the difference $D_{MKS_{r(11,13,17)}} = 0.5$ is not a random chance effect, thus at least two out of the three areas 11, 13 and 17 have statistically significantly different distributions.

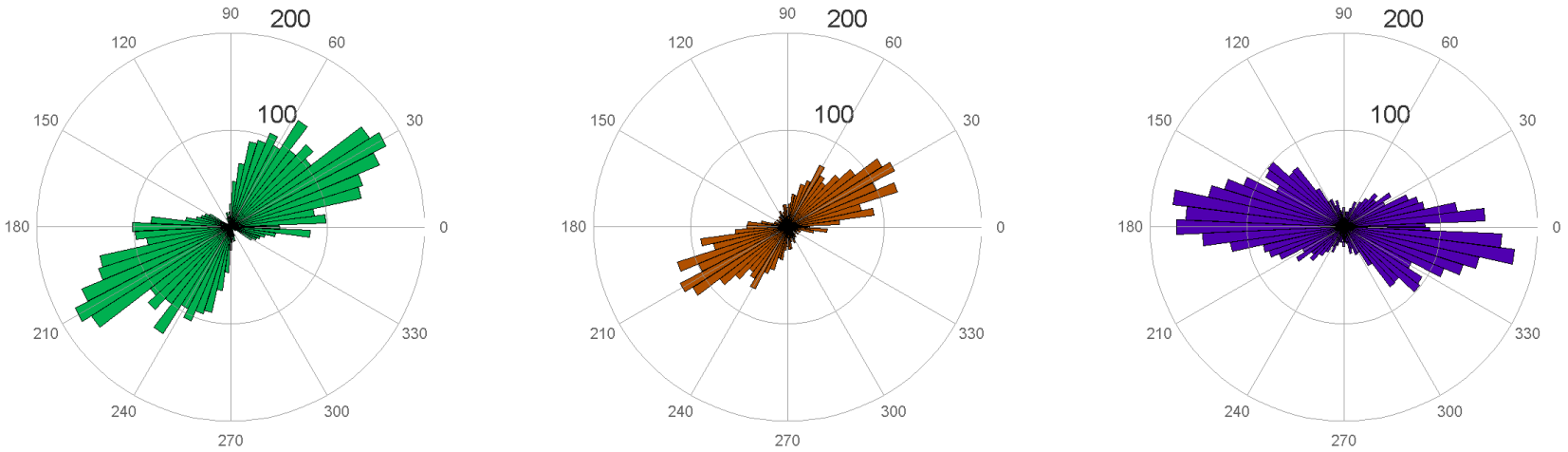
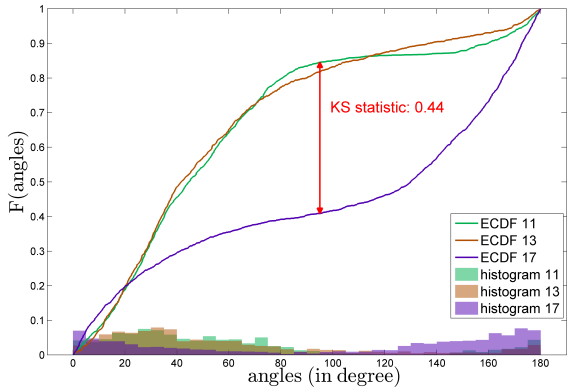
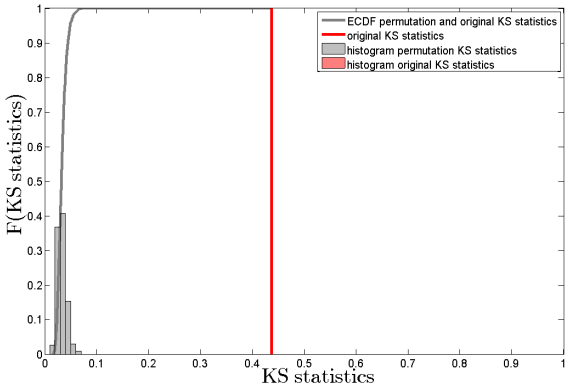


Figure 6.7 ***Rose plots representing the orientation and quantity of unweighted fault lines in three 48 $\times$ 48 km^2 areas.*** *Rose plots are angle histograms. The bin size is five degree and the frequency is visualized by the circles up to 200. Note, the fault lines are mirrored, and thus bins opposite to each other represent the same fault lines. The presented fault lines are from the 11th area (green), 13th area (brown) and 17th area (purple) from figure 6.5.*

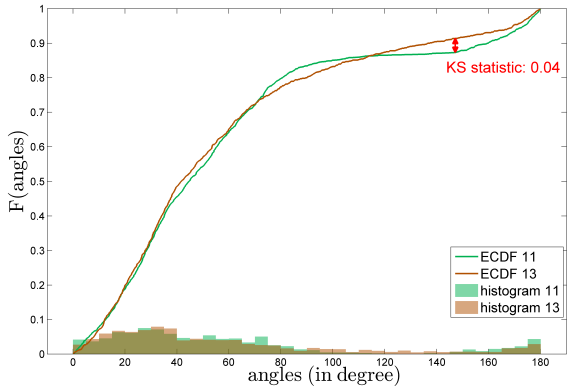


Ecdfs of orientation for area 11, 13 and 17.

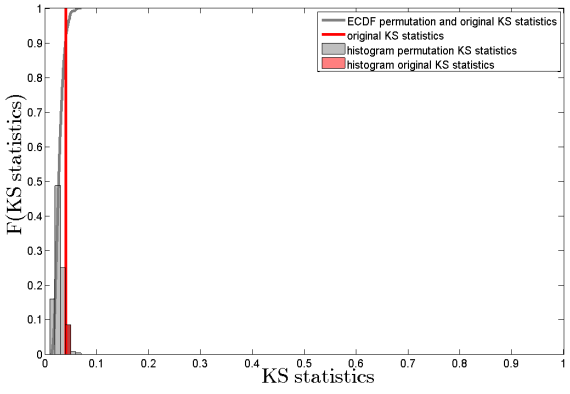


Sampling distribution for area 11, 13 and 17.

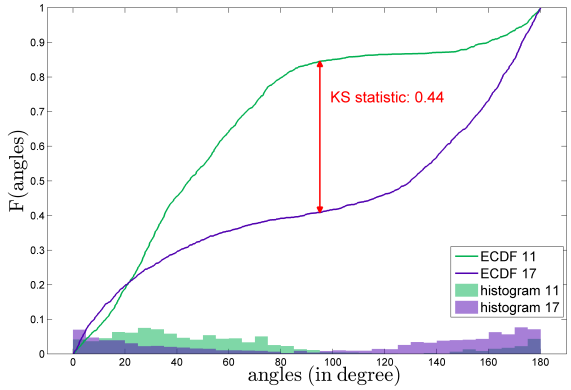
Figure 6.8 ***MKS for unweighted fault lines in three 48 × 48 km^2 areas.*** *The MKS for the unweighted angles results in a statistically significant difference.*
left: *The maximum vertical distance between the empirical distributions function of fault line orientation is D_{MKS_r} = 0.44. The histogram represents the relative frequency of the angles of these fault lines. Histograms can be thought to be unwrapped roseplots.*
right: *No D_{MKS_p} is greater than D_{MKS_r} = 0.44 ($p < 0.001$. This supports the decision to reject H_0 and conclude that at least two out of three areas have different distributions.*



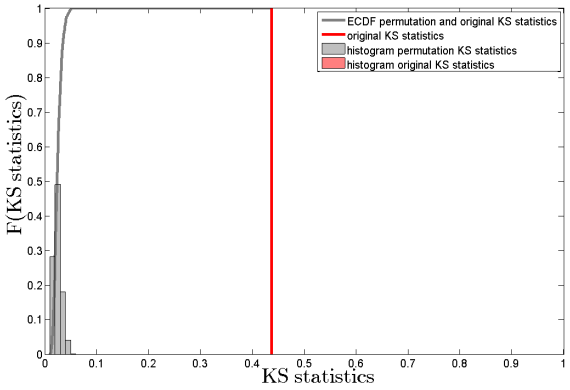
Ecdf of orientation for area 11 (green) and 13 (brown).
D_{2KS_r} = 0.04 (p > 0.05).
H_0 is not rejected. There is support that the areas have the same distributions.



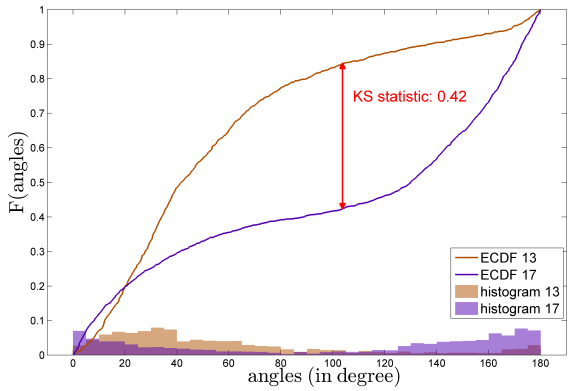
Sampling distribution for area 11 and 13.
Some D_{2KS_p} are greater than $D_{2KS_{r(11,13)}}$ = 0.04, thus the difference is statistically not significant (p > 0.07) and the result of the classical 2KS is confirmed.



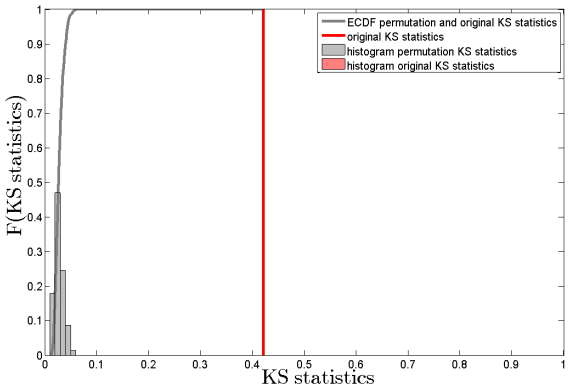
Ecdf of orientation for area 11 (green) and 17 (purple).
D_{2KS_r} = 0.44 (p < 0.001).
Rejection of H_0 and conclusion that the areas have different distributions.



Sampling distribution for area 11 and 17.
No D_{2KS_p} is greater than $D_{2KS_{r(11,13)}}$ = 0.44, thus the result of the classical 2KS is confirmed.

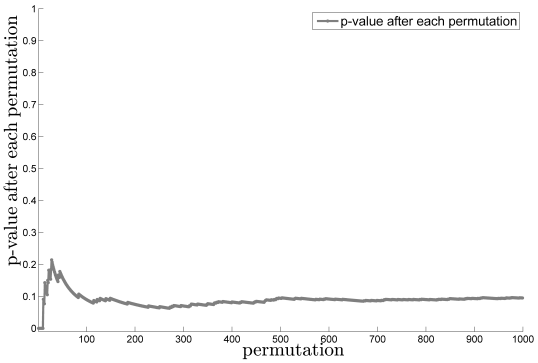


Ecdf of orientation for area 13 (brown) and 17 (purple).
D_{2KS_r} = 0.42 (p < 0.001).
Rejection of H_0 and support that the areas have different distributions.

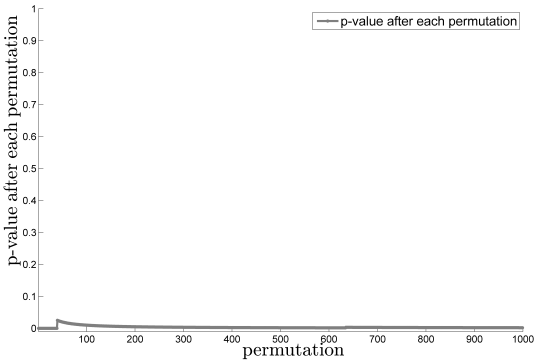


Sampling distribution for area 13 and 17.
No D_{2KS_p} is greater than $D_{2KS_{r(11,13)}}$ = 0.42, thus the result of the classical 2KS is confirmed.

Figure 6.9 ***Classical and permutation 2KS for unweighted fault lines in three 48 × 48 km^2 areas.***
left: Classical 2KS: *The maximum vertical distance between the empirical cumulative distribution functions is the test statistic D_{2KS_r}. In two of three comparisons of unweighted angles, the test statistic is greater than the critical value at α 0.05. In these cases the H_0 is rejected. The histogram represents the relative frequency of the angles of these fault lines. Histograms can be thought to be unwrapped roseplots.*
right: Permutation 2KS: *The red line indicates the test statistic D_{2KS_r}. In contrast to the bins of the permutation test statistics, the bin for the test statistic D_{2KS_r} in the middle and bottom histogram is too small for visualization.*



The p-value for the unweighted angles fluctuates in the first permutations before converging to $p \approx 0.10$.



The p-value for the weighted angles fluctuates in the first permutations before converging to ≈ 0.001.

Figure 6.10 ***2KS p-value after each permutation unweighted and weighted fault lines in area 11 and 13.***

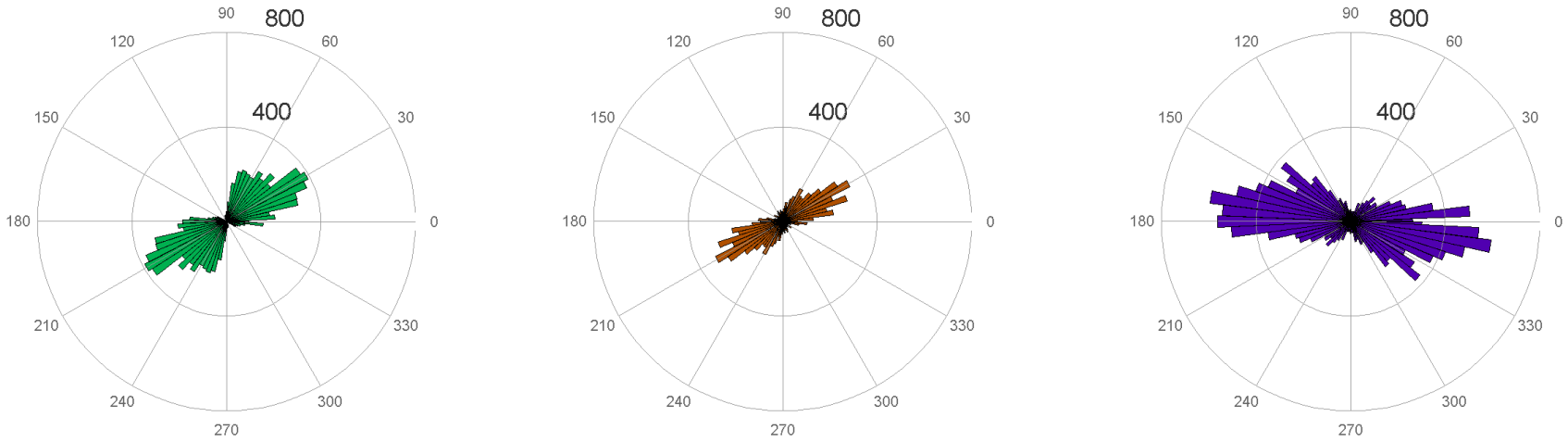
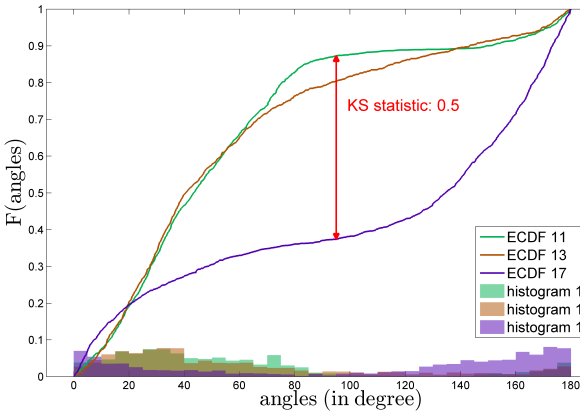
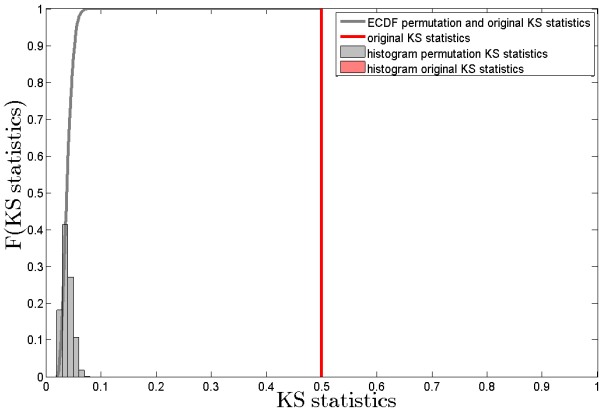


Figure 6.11 ***Rose plots representing the orientation and quantity of weighted fault lines in three 48 × 48 km² areas.*** *The presented fault lines are from the 11th area (green), 13th area (brown) and 17th area (purple) from figure 6.5. The angles were weighted with the length of the fault line segment. The weight is a multiple of 100 m of the line segment's length, so that a line segment of 443 m is described by 5 angle elements. In other words, a bin of 400 includes 400 line segments each one of the length up to 100 m.*

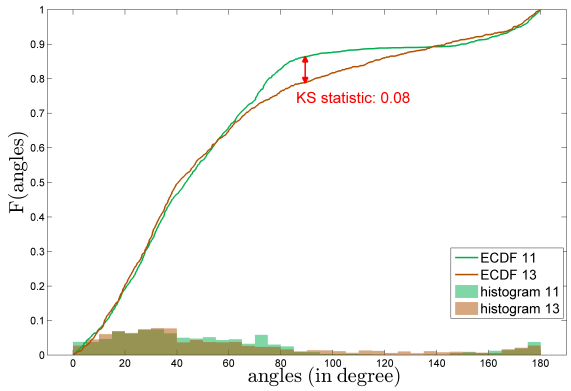


Ecdf of orientation for area 11 (green), 13 (brown) and 17 (purple).

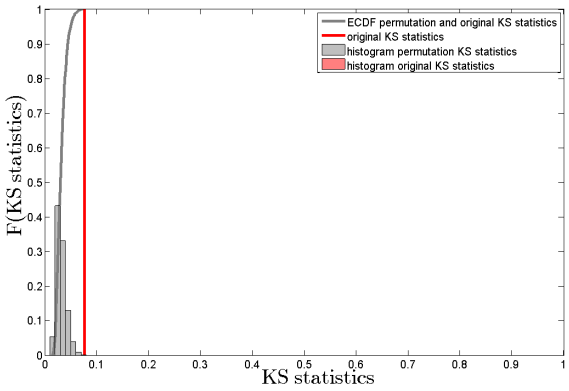


Sampling distribution for area 11, 13 and 17.

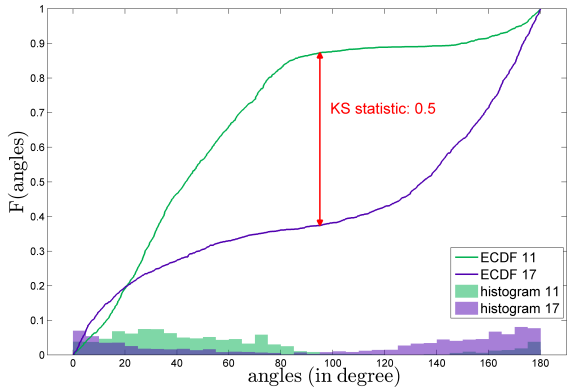
Figure 6.12 ***MKS for weighted fault lines in three 48 $\times$ 48 km^2 areas.*** *The MKS for the weighted angles results in a statistically significant difference.*
left: *The maximum vertical distance between the empirical distributions function is $D_{MKS_r} = 0.5$. The histogram represents the frequency of the angles of the fault lines for each area. Here, histograms can be thought to be unwrapped roseplots.*
right: *The red line indicates the test statistic $D_{2KS_r} = 0.5$. No D_{2KS_p} is greater than $D_{MKS_r} = 0.5$, thus $p < 0.001$. This supports the decision to reject H_0 and conclude that at least any two out of three distributions are statistically significantly different. In contrast to the bins of the permutation test statistics, the bin for the test statistic D_{2KS_r} is too small for visualization.*



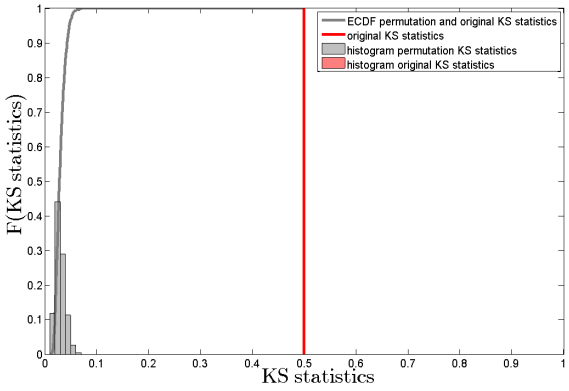
Ecdf of orientation for area 11 (green) and 13 (brown). D_{2KS_r} = 0.08 ($p < 0.001$). H_0 is rejected and there is support that the areas have different distributions.



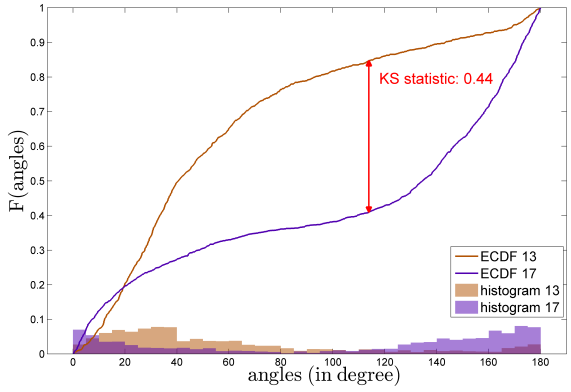
Sampling distribution for area 11 and 13. Some D_{2KS_p} are greater than D_{2KS_r} = 0.08. Here, the p-value is changing in the first permutations but converges fast to $p < 0.001$, thus the result of the classical 2KS is confirmed.



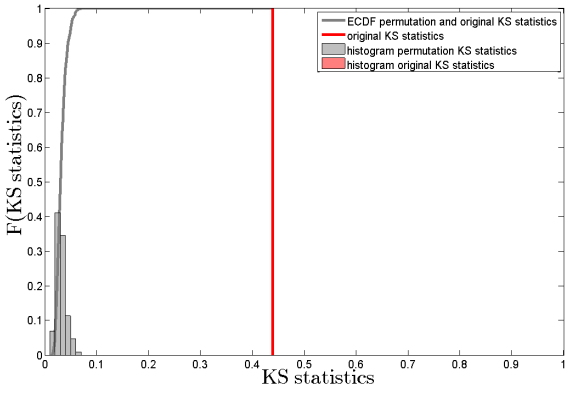
Ecdf cumulative distribution functions for area 11 (green) and 17 (purple). D_{2KS_r} = 0.5 ($p < 0.001$). H_0 is rejected and there is support that the areas have different distributions.



Sampling distribution for area 11 and 17. No D_{2KS_p} is greater than D_{MKS_r} = 0.5, thus $p < 0.001$. The result of the classical 2KS is confirmed.



Ecdf of orientation for area 13 (brown) and 17 (purple). D_{2KS_r} = 0.44 ($p < 0.001$). H_0 is rejected and there is support that the areas have different distributions.



Sampling distribution for area 13 and 17. No D_{2KS_p} is greater than D_{MKS_r} = 0.5, thus $p < 0.001$. The result of the classical 2KS is confirmed.

Figure 6.13 ***Classical and permutation 2KS for weighted fault lines in three 48 × 48 km² areas.*** *The difference between area 11 and 13 is the only one that is not statistically significant.* ***left: Classical 2KS:*** *The test statistic D_{2KS_r} is the maximum vertical distance. The histogram represents the frequency of the angles of the fault lines for each area. Here, histograms can be thought to be unwrapped roseplots.*
right: Permutation 2KS: *The red line indicates the test statistic D_{2KS_r}. In contrast to the bins of the permutation test statistics, the bin for the test statistic D_{2KS_r} is too small for visualization.*

Kolmogorov-Smirnov classical test for two samples (2KS) including the permutation test. The highest test statistics is $D_{2KS_{r(11,17)}} = 0.5$ between area 11 and 17, closely followed by $D_{2KS_{r(13,17)}} = 0.44$ for area 13 and 17. Between area 11 and 13 the test statistic is only $D_{2KS_r} = 0.08$ illustrated in Figure 6.13.

The classical 2KS result indicates that the D_{2KS_r} for all three comparisons are statistically significant ($p < 0.001$). For all three comparisons H_0 is rejected and the conclusion is that there is a statistically significant difference in the distributions of fault line orientations between all pairwise compared areas.

The permutation 2KS results support the decision that these three differences D_{2KS_r} are not random. The D_{2KS_p} is for the comparison between area 11 and 17, as well as 13 and 17 never higher than the $2KS_{r(11,17)} = 0.5$ and $D_{2KS_{r(13,17)}} = 0.44$ shown in Figure 6.13. Only for the comparison between area 11 and 13, there are a few D_{2KS_p} greater than D_{2KS_r}. Here, the p-value is changing in the first permutations but converges fast to $p < 0.001$.

6.6 Summary and Discussion

This case study used the GNS QMAP to analyse the orientation of fault lines on the South Island in New Zealand. The classical 2KS, permutation 2KS and MKS were trialled as ways to analyse the difference between areas.

Kolmogorov-Smirnov permutation test for multiple samples (MKS). The new developed MKS test for differences in the distribution of orientation of fault lines between any of m areas. In this study, it could have been used to compare all eighteen 48×48 km^2 areas at once. Because of computational restriction, it was not possible to randomize all angles describing the orientation of fault line segments in all eighteen areas. However, if we assume, H_0 would have been not rejected with the MKS, and the difference between the observed ecdfs expressed by D_{MKS_r} is implied to be due to random chance rather than the actual difference between the areas, then no 2KS test would have been necessary. For this example, one MKS test would have replaced 153 $\left(\frac{18 \times 17}{2}\right)$ 2KS tests. This reduces the required computational time. In this example, the single MKS result gives a fast overview compared to 153 2KS results.

Number of Permutations. In almost all comparisons of two areas with the permutation 2KS, the p-value after each of the 999 permutation was $p < 0.001$. In these cases, the number of permutations could have been decreased as there was no variability. In future analyses the number of the permutations can be adjusted to the variability of the p-value. If the p-value appears to converge, there are no further permutations necessary.

Kolmogorov-Smirnov classical and permutation test for two samples (2KS) for unweighted and weighted version. It was not expected that almost all areas have statistically significant different distributions for both, the unweighted and weighted angles. For the unweighted version, almost all of the pairwise 2KS classical and permutation test results between the eighteen 48×48 km^2 areas were statistically significant different. The only exception is the comparison between area 11 and 13. For the weighted version, all pairwise tested distributions were statistically significantly different. It is possible that the weighting with a multiple of 100 m was to coarse. A weighting for each 1 m could have led to another result. Computational power did not allow this approach. The question arises, whether the result would have been different if weighting had been based not on the length of the line segments but with other attributes. One idea is to weight the angles with the accuracy of each mapped fault line. Geologists described the accuracy of the line segment with *accurate*, *approximate*, *concealed* or *inferred* (described in more detail in Section 2.10). More weight would be assigned to *Accurate* fault lines than to *inferred* lines.

Angle Spectrum Cut Off. The major advantage of testing the orientation fault lines in areas with the distribution instead of the mean is that the position where to cut the spectrum of angles does not influence the result. For example, the test statistics D_{2KS} and D_{MKS} would have been the same for the spectrum from 0° to 180°and the

spectrum 45° to 225°. The reason is that the maximum vertical distance in the ecdfs, and the decision based on it, are independent from the angle at which it occurs. For example, the D_{MKS_r} = 0.44 at 75° for the spectrum from 0° to 180° would be D_{MKS_r} 0.44 at 120° for the spectrum from 45° to 225°. The decision whether to reject H_0 is not influenced by the spectrum cut off.

Differences in shape, spread or median? Despite the statistically significant results, we do not know whether the distributions differ in median, variability or shape. This information would provide the opportunity to specifically describe the differences in the orientation of fault lines between areas. Substantial differences in shape, spread or median lead to a small p-value for both the 2KS and MKS test. The advantage of the 2KS and MKS is the three-in-one testing (shape, spread, median). In contrast, the advantage of more specific tests, such as the Mann-Whitney test, that is mostly sensitive to changes in the median, allows a more specific interpretation. However, for the Mann-Whitney and any other test focusing on the central tendency the spectrum influences the result. For example, the three angles 10°, 50° and 120° result in the median 50°. In the spectrum 40° to 220° the same angles are transformed to 190°, 50° and 120°. The median would be 120° instead of 50°. As suggested by the analysis, the mean and the median are not useful to describe the central tendency of the orientation of fault lines.

MKS Critical Values. At the current stage the MKS is a permutation test. For future research I suggest to extend the MKS with critical values so that the D_{MKS_r} can directly be compared. These critical values can be developed similar to the critical values for the 2KS test.

MKS Type 1 error. Besides the advantage of the MKS of simultaneous testing of more than two distributions, the MKS controls the probability of a type I error to be the level of significance originally chosen by the user. There is no inflation of type 1 error for the MKS compared to the increase of inflation for the 2KS with an increasing number of pairwise tests.

Chapter 7

Concluding comments

In this thesis, I have attempted to quantify the pattern of fault lines on the South Island of New Zealand. In particular, I have applied Box-Counting method to see whether the pattern could be described as fractal and, if so, what was its estimated fractal dimension. I have also adapted the Kolmogorov-Smirnov test – a common statistical technique for comparing distributions – to see whether distribution of orientation of fault lines is independent of location. The adapted test is a permutation test which allows the user to compare several distributions simultaneously.

Application of Box-Counting methods to the artificial fractal object called the Koch Curve has demonstrated that the fractal dimension evaluation is highly dependent on the choices made by the modeler. For the Koch Curve, the estimated fractal dimension varied from 0.73 to 2.0, depending on interpolation, magnification factor, shape of boxes, the position of the vector data on the grid, and the method to estimate the fractal dimension. For the fault lines, it was found that the estimated fractal dimension decreases with an increase of scale: this is consistent with the findings of the Japanese study, that motivated the current work. One may extrapolate from these findings that the fractal characteristics of different fault systems are quite similar. Quantifying the fractal characteristics in one system can be helpful for understanding the fault systems in other countries.

The adapted Kolmogorov-Smirnov test has shown that the distribution of orientation of fault lines differs between different areas of the South Island. Although the difference is statistically significant, its practical significance can only be understood in the light of its geological and seismological implications.

For the fault lines on the South Island, Box-Counting does not return the power-law relationship that is expected for mono-fractal objects. This leads me to agree with Gillespie, Howard, Walsh and Watterson[37] that Box-Counting is too insensitive to characterise different features of a fractal objects, and further with Hall and Wood[43] who point out that estimators of fractal dimension are critically different between the mathematical formulaes they employ. For future research, I would recommend the use of fractal methods that can be adjusted to irregular shaped object, rather than using "Box-Counting" where fractal objects are forced into quadrangles. Moreover, further research of fault line orientation should help clarify details in differences between geological areas.

With the increasing abilities of modern technology to monitor earthquakes and reveal the existence of previously unknown fault lines, there is still a need for further quantification of fault line patterns around the globe and within each fault system in particular. It is only as we build a better picture of fault line placement and its orientation that we can move towards understanding the processes behind the formation of any particular fault system. This will greatly help with the identification of missing faults thereby leading to improved earthquake risk management and safety procedures of the future.

Appendix A

Illustrations

Illustration A.1 ***Calculation of Cartesian Coordinates for the Koch Curve***

In Section 4.1 the conversion of vector data to raster data is explained. For calculating the Cartesian coordinates I transform the relative degrees of the angles to absolute degrees related to the origin by adding the relative angle to the preceding absolute angle. Then I calculate Cartesian coordinate values based on the line segment length and angle related to the origin. Starting from the left these coordinate values are added to the preceding point of the Koch Curve to get the final coordinates with

$$\begin{aligned} X_{i,j} &= X_{j-1} + (\tfrac{1}{3})^i \ cos \ a_j \\ \text{and } Y_{i,j} &= Y_{j-1} + (\tfrac{1}{3})^i \ sin \ a_j \ , \end{aligned}$$

where $X_{i,j}$ and $Y_{i,j}$ are the coordinate values for the jth point of the ith Koch Curve iteration, a_i is the direction of the vector given as angle with respect to the origin, and $(\frac{1}{3})^j$ is the magnitude given as line segment length in units . The $X_{i,j}$ and $Y_{i,j}$ Cartesian coordinates provide the magnitude and the direction of the line segments.

Illustration A.2 ***Constructing Raster***

To keep the relationship between the vector format and raster format I construct a Matlab `MapRasterReference` object shown in figure A.1. The inputs are based on Cartesian coordinates and the magnification factor. For the Koch Curve, the inverse of the magnification factor is the grid's box edge length in units and this defines how large the boxes should be. By increasing the magnification factor, the number of boxes increases and therefore the box size decreases.

The size of the boxes influences the number of boxes, and thus the resolution. The higher the resolution, the higher the number of boxes used and therefore the smaller the size of an area described by one single box. A higher resolution leads to a more detailed description of the pattern. However, computational capacity limits the resolution and we often have to be satisfied with a lower quality.

For demonstrating the process of converting vector into raster data, I take a Koch Curve in the second iteration, a magnification factor of $m = 5$ and a grid with square boxes. The frame of the grid is defined by the minimum and maximum X and Y values. I determine these values with the Matlab functions `min()` and `max()` and get

$$\begin{aligned} & min(X) = 0 && max(X) = 1 \\ \text{and} \quad & min(Y) = 0 && max(Y) = 0.289 \ . \end{aligned}$$

The box edge length e is calculated dividing the longer frame edge by the magnification factor m with

$$e = \frac{max\ (\ max(X)-min(X)\ ,\ max(Y)-min(Y)\ )}{m} = \frac{1}{5} = 0.2 \text{ units} .$$

The number of the boxes of the grid is calculated with

$$n_R = \left\lceil \frac{max(Y)-min(Y)}{e} \right\rceil = \left\lceil \frac{0.289-0}{0.2} \right\rceil = 2$$
$$\text{and} \quad n_C = \left\lceil \frac{max(X)-min(X)}{e} \right\rceil = \left\lceil \frac{1-0}{0.2} \right\rceil = 5 ,$$

where n_R is the number of rows and n_C is the number of columns. In the `MapRasterReference` these values are stored as `RasterSize: [2 5]`. To get square boxes, the grid width W and height H is increased to a multiple of the box edge length with

$$w = n_C \times e = 5 \times 0.2 = 1$$
$$\text{and} \quad h = n_R \times e = 2 \times 0.2 = 0.4 .$$

For the Koch Curve the raster height increases from 0.289 to 0.4. These values are in the `MapRasterReference` as `RasterWidthInWorld` and `RasterHeightInWorld`. I determine the Cartesian coordinates for the frame of the grid with

$$X_l = min(X) = 0$$
$$\text{and} \quad X_r = X_l + w = 1$$
$$\text{and} \quad Y_b = min(Y) = 0$$
$$\text{and} \quad Y_t = Y_b + h = 0.4 ,$$

where X_l, X_r, Y_b, Y_t are the left, right, bottom and top border of the grid, respectively. In the `MapRasterReference` they are called `XLimWorld` and `YLimWorld`.

`DeltaX` and `DeltaY` in the `MapRasterReference` define the box width and height. In our example both are 0.2 units shown in figure A.1 and confirm that the boxes are square. `DeltaX` and `DeltaY` are positive signed indicating that the column and row number increases when X and Y increases. This is confirmed in the Matlab `MapRasterReference` in figure A.1 with `ColumnsStartFrom: 'south'` and `RowsStartFrom: 'west'`. This congruency eases the comprehension of the conversion from vector to raster.

The Cartesian coordinates of the points of the Koch Curve are converted into raster coordinates with the Matlab `MapRasterReference` and its related function `R.worldToInstrinsic`, where `World` stands for the Cartesian Coordinates and `Instrinsic` for the raster coordinates. The raster coordinates indicate the column and row of boxes that are occupied by a point. For example, the highest point of the Koch Curve in the second iteration is p_9 illustrated in figure 4.3. The `world` coordinates of this point are p_9 (0.5,0.289) and the `intrinsic` coordinates determined with `R.worldToInstrinsic` are p_9 (3.0,1.9). So far, I determined the information for an empty grid and I converted the Cartesian Coordinates in vector format into raster coordinates. This raster can be reflected by a Matlab matrix.

To index the rows and column in the matrix the intrinsic raster coordinates need to be rounded to integer, in our example p_9 (3,2). The intrinsic coordinates `XLimIntrinsic` and `YLimIntrinsic` start from 0.5 so that the minimum rounded intrinsic coordinate value is 1 used as index for the first row or first column. Note, in contrast to the raster start *south-west*, matrix indexing starts *north-west* and so for the matrix here, the numbers can be thought of as being upside-down (figure 4.3).

This procedure results in a raster for the Koch Curve only determined by the start and end points of the line segments rather than the line segments themselves. For a vector polyline of the Koch Curve in the first iteration these are only four points. These four points describe the Koch Curve in a raster with boxes of the length $\frac{1}{5}$ units well, but if having smaller boxes of the length $\frac{1}{9}$ units, not all boxes crossed by the polyline are identified shown in figure 4.7a on the right side.

Illustration A.3 ***Interpolation***

The Matlab built-in function `doInterpm` computes the difference between the two X values and the difference between the two Y values of the points for each line segment. If the longer of these two distances exceeds the interpolation length in units l_U, the line segment between the points is filled with additional points. You might expect that every box a line segment travels through would be identified as being occupied as the distance between the (X, X) coordinates of two sequential points is for l_R = 0.90 always smaller than the box edge length.

For the Koch Curve in the first iteration on the right side in figure 4.7a we have p_2 (0.333,0) and p_3 (0.5,0.289). The distance d between the X and Y values of these two points is

$$\begin{aligned} d_{(x_i,x_{i+1})} &= |\, x_i - x_{i+1} \,| \\ d_{(x_2,x_3)} &= |\, x_2 - x_3 \,| \quad = \quad |\, 0.5 - 0.333 \,| \quad = \quad 0.167 \text{ units} \end{aligned}$$

$$\begin{aligned} \text{and} \quad d_{(y_i,y_{i+1})} &= |\, y_i - y_{i+1} \,| \\ d_{(y_2,y_3)} &= |\, y_2 - y_3 \,| \quad = \quad |\, 0 - 0.289 \,| \quad = \quad 0.289 \text{ units} . \end{aligned}$$

The function `doInterpm` takes the longer distance as basis for the calculation of additional points. The number of additional points n_P with this function is

$$n_P = \left\lfloor \frac{max(\delta_{(x_2,x_3)}, \delta_{(y_2,y_3)})}{l_U} \right\rfloor = \left\lfloor \frac{max(0.167, 0.289)}{0.1} \right\rfloor = 2 .$$

The function `doInterpm` adds these two points evenly spaced on the line, and therefore boxes can be missed even if they occupy a line segment equal or longer than the interpolation length.

In contrast to checking the distances between the (X, X) coordinates as `doInterpm` does, the method `doPythagoras` determines the length of the line segment itself and interpolated if it was longer than l_U, in our case 0.1 units. In our example, the line segment length is

$$\begin{aligned} l_S &= \sqrt{d^2_{(x_2,x_3)} + d^2_{(y_2,y_3)}} \\ &= \sqrt{0.167^2 + 0.289^2} \quad = \quad 0.334 \text{ units} , \end{aligned}$$

where l_S is the length of the line segment. The number of additional points n_P is

$$\begin{aligned} n_P &= \left\lfloor \frac{l_S}{l_U} \right\rfloor \\ &= \left\lfloor \frac{0.334}{0.1} \right\rfloor \quad = \quad 3 \end{aligned}$$

To clarify, the method also detects some boxes that are occupied by a line segment shorter than l_U. This is the

case for the blue box in figure 4.7c The frame of the box four from the left and two up has the coordinates

$$\begin{array}{lll} & X_l = \frac{3}{9} \\ \text{and} & X_r = \frac{4}{9} \\ \text{and} & Y_b = \frac{1}{9} \\ \text{and} & Y_t = \frac{2}{9} \, , \end{array}$$

where X_l, X_r, Y_b, Y_t are the left, right, bottom and top border of the box, respectively. The coordinates of the line segment travelling through the box are for the start point p_s (0.398,0.111) and for the endpoint p_e (0.444,0.193) determined with the code `trimPolylineToQuadrangle` (appendix D.4). The length of this line segment is

$$\begin{array}{rcl} l_S & = & \sqrt{d^2_{(x_s,x_e)} + d^2_{(y_s,y_e)}} \\ & = & \sqrt{|\, x_s - x_e \,|^2 + |\, x_s - x_e \,|^2} \\ & = & \sqrt{|\, 0.398 - 0.444 \,|^2 + |\, 0.111 - 0.193^2} \quad = \quad 0.093 \text{ units} \, . \end{array}$$

`doPythagoras` identified the box as occupied even that the line segment with l_S = 0.093 units was smaller than the interpolation length l_U = 0.1 units.

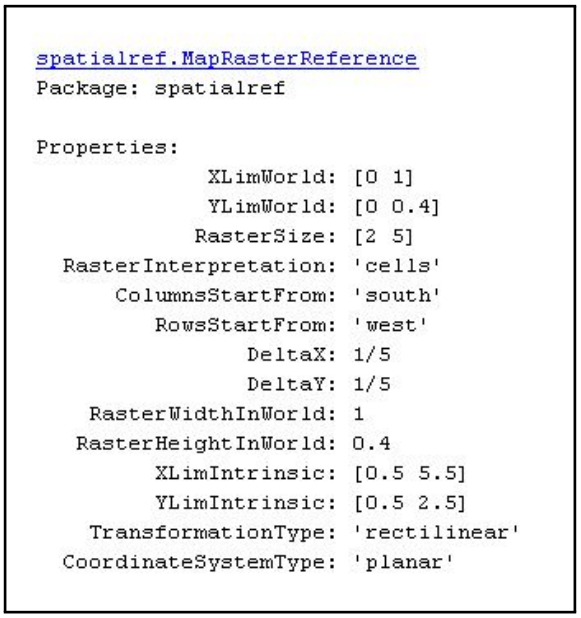


Figure A.1 ***Matlab `MapRasterReference` object for storage of relationship between Cartesian Coordinates and Raster Coordinates.*** *A Matlab `MapRasterReference` object summarizes the most important parts of the relationship between Cartesian coordinates and raster coordinates. It shows the Cartesian limits with `XlimWorld` and `YlimWorld` and the raster limits with `XlimIntrinsic` and `YlimIntrinsic`. The rounded raster coordinates are the indices for boxes with row and column number. The `MapRasterReference` defines from which point the boxes are counted, here it is south-west given as `ColumnsStartFrom: 'south'` and `RowsStartFrom: 'west'`. `DeltaX` and `DeltaY` have the same value indicating that the boxes of the raster are squares. `'rectilinear'` means that the intrinsic X and world X axes align and the intrinsic Y and world Y axes align.*

Appendix B

Figures

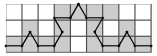
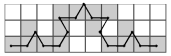
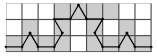
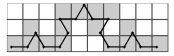
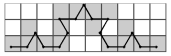
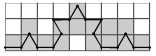
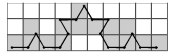
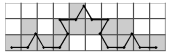
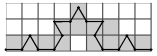
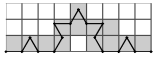
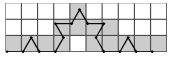
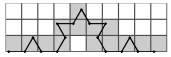
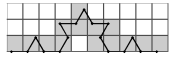
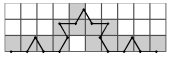
$h_R = \frac{0}{10}, v_R = \frac{4}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{1}{10}, v_R = \frac{4}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{2}{10}, v_R = \frac{4}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{3}{10}, v_R = \frac{4}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{4}{10}, v_R = \frac{4}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 
$h_R = \frac{0}{10}, v_R = \frac{3}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{1}{10}, v_R = \frac{3}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{2}{10}, v_R = \frac{3}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{3}{10}, v_R = \frac{3}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 	$h_R = \frac{4}{10}, v_R = \frac{3}{10}$ $m = 9$ $N = 16$ $fd_{LR} =$ **1.26** 
$h_R = \frac{0}{10}, v_R = \frac{2}{10}$ $m = 9$ $N = 15$ $fd_{LR} =$ **1.23** 	$h_R = \frac{1}{10}, v_R = \frac{2}{10}$ $m = 9$ $N = 15$ $fd_{LR} =$ **1.23** 	$h_R = \frac{2}{10}, v_R = \frac{2}{10}$ $m = 9$ $N = 15$ $fd_{LR} =$ **1.23** 	$h_R = \frac{3}{10}, v_R = \frac{2}{10}$ $m = 9$ $N = 15$ $fd_{LR} =$ **1.23** 	$h_R = \frac{4}{10}, v_R = \frac{2}{10}$ $m = 9$ $N = 15$ $fd_{LR} =$ **1.23** 
$h_R = \frac{0}{10}, v_R = \frac{1}{10}$ $m = 9$ $N = 13$ $fd_{LR} =$ **1.17** 	$h_R = \frac{1}{10}, v_R = \frac{1}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{2}{10}, v_R = \frac{1}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{3}{10}, v_R = \frac{1}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{4}{10}, v_R = \frac{1}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 
$h_R = \frac{0}{10}, v_R = \frac{0}{10}$ $m = 9$ $N = 13$ $fd_{LR} =$ **1.17** 	$h_R = \frac{1}{10}, v_R = \frac{0}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{2}{10}, v_R = \frac{0}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{3}{10}, v_R = \frac{0}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 	$h_R = \frac{4}{10}, v_R = \frac{0}{10}$ $m = 9$ $N = 14$ $fd_{LR} =$ **1.2** 

Figure B.1 *Koch Curve translation for all 25 positions*

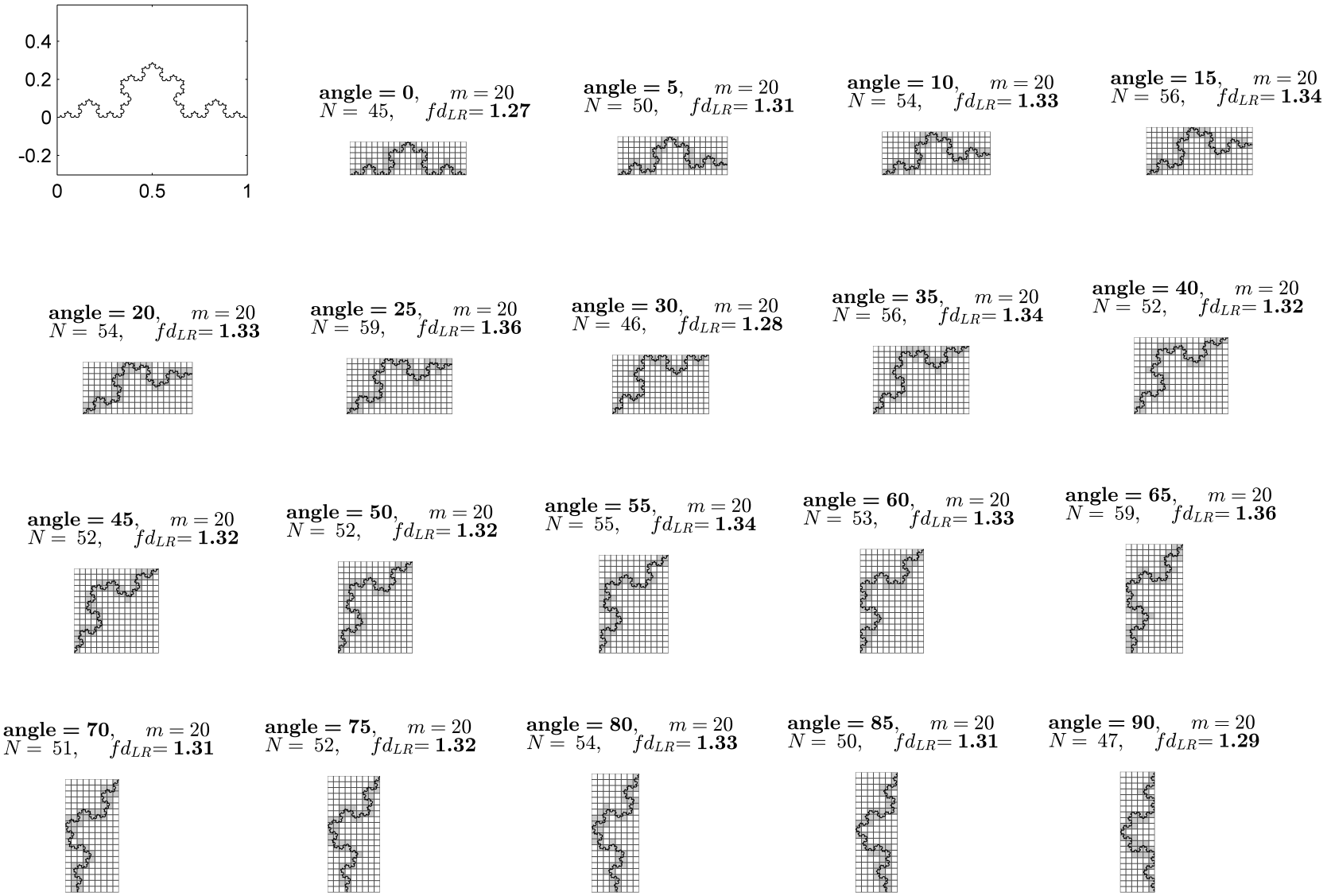


Figure B.2 *Koch Curve rotation with square boxes for all 19 angles*

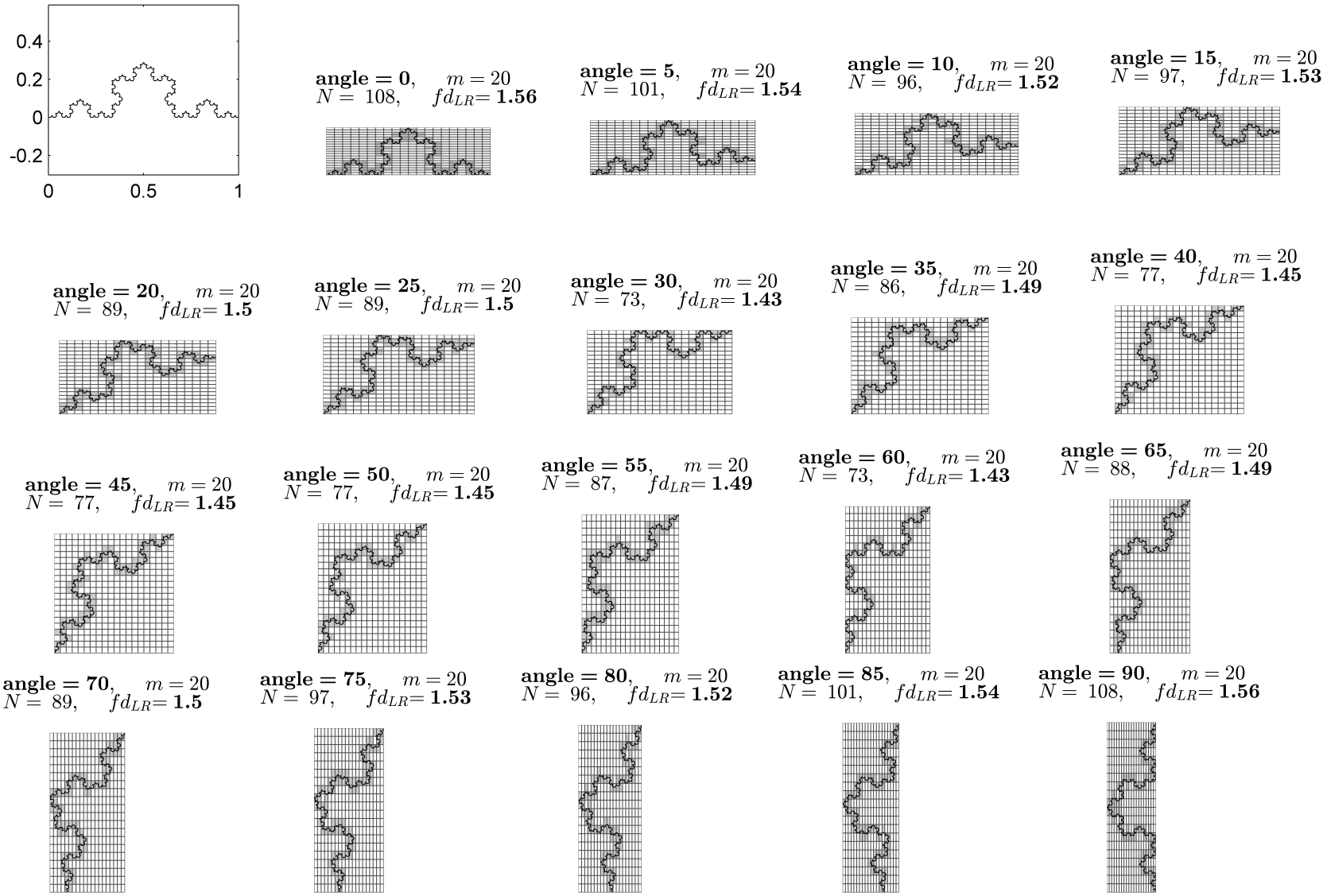


Figure B.3 *Koch Curve rotation with rectangular boxes for all 19 angles*

Appendix C

Tables

m	N	fd_{FD}	fd_{MD}	fd_{LR}	difference fd_{MD} and fd_{LR}
10	16	1.46	1.46	1.20	-0.26
20	44	1.35	1.42	1.26	-0.16
30	76	**1.35**	1.35	1.27	-0.07
40	112	**1.48**	1.41	1.28	-0.13
50	**156**	**1.11**	**1.32**	**1.29**	-0.03
60	**191**	1.00	**1.06**	**1.28**	**0.22**
70	**223**	1.40	**1.19**	**1.27**	0.08
80	269	1.15	1.28	1.28	-0.01
90	308	1.10	1.13	1.27	0.15
100	346		1.10	1.27	0.17

	$fd_{GM(FD)}$	$fd_{GM(MD)}$	$fd_{GM(LR)}$
0.00	**1.27**	**1.27**	**1.27**

Table C.1 ***Influence of different calculation methods on estimated local and global fractal dimension for the Koch Curve.***

	figure	Koch Curve iteration	box form	magnification factor	magnification factor based on the size of	box size	interpolation method	interpolation length l_R	multiple matrices or single matrix	position	count	log ratio	range of df
a)	figure 4.7	1st	squares	9	actual pattern	$\frac{1}{9}$	**none** **doInterpm** **doPythagoras**	0.90	multiple	constant	5 11 12	0.73 1.09 1.13	0.40
b)	figure 4.8	2nd	squares	20	actual pattern	$\frac{1}{20}$	**none** **doInterpm** **doPythagoras**	0.99	multiple	constant	17 31 31	0.95 1.15 1.15	0.20
c)	figure 4.6	**2nd** **3rd** **4th**	squares	9	actual pattern	$\frac{1}{9}$	doPythagoras	0.99	multiple	constant	13 14 15	1.17 1.20 1.23	0.07
d)	figure 4.9	2nd	squares	25	actual pattern	$\frac{1}{25}$	doPythagoras	**0.30** **0.99**	multiple	constant	47 41	1.20 1.15	0.05
e)	figure 4.10	2nd	squares	$2^5 = 32$	actual pattern	$\frac{1}{32}$	doPythagoras	0.99	**multiple** **single matrix based on m 2^7**	constant	56 66	1.16 1.21	0.06
f)	figure 4.11	2nd	squares	5 8	**actual pattern** **2^n matrix**	$\frac{1}{5}$ $\frac{1}{8}$	doPythagoras	0.99	multiple	constant	6 6	1.11 0.86	0.25
g)	figure 4.12	2nd	**squares** **rectangles**	5	actual pattern	$\frac{1}{5}$	doPythagoras	0.99	multiple	constant	6 12	1.11 1.54	0.43
h)	figure 4.14	2nd	squares	9	actual pattern	$\frac{1}{9}$	doPythagoras	0.99	multiple	**translation (original position)** **translation ($h_R = \frac{0}{10}$, $v_R = \frac{4}{10}$)**	13 16	1.17 1.26	0.09
i)	figure 4.15	4th	squares	20	actual pattern	$\frac{1}{20}$	doPythagoras	0.99	multiple	**rotation (0 degree)** **rotation (65 degrees)**	45 59	1.27 1.36	0.09
j)	figure 4.16	4th	rectangles	20	actual pattern	$\frac{1}{20}$	doPythagoras	0.99	multiple	**rotation (0 degree)** **rotation (60 degree)**	108 73	1.56 1.43	0.13

Table C.2 ***Range in the estimated fractal dimension for the Koch Curve.*** *The range in the Box-Counting fractal dimension between different approaches in the two steps converting data and calculating the fractal dimension can be up to 0.40, e.g. between undertaking an interpolation of the vector data or not.*

magnification factor	box edge length in metre based on area 96,000 m x 96,000 m		number of covered boxes fd_{LR}
5	**19,200.00**	**25**	**2.00**
10	9,600.00	97	1.99
11	8,727.27	111	1.96
12	8,000.00	130	1.96
14	6,857.14	173	1.95
16	6,000.00	215	1.94
19	5,052.63	280	1.91
24	4,000.00	404	1.89
32	3,000.00	601	1.85
48	2,000.00	1,021	1.79
96	1,000.00	2,335	1.70
110	872.73	2,752	1.68
120	**800.00**	**2,961**	**1.67**
140	685.71	3,506	1.65
160	600.00	4,000	1.63
192	500.00	4,808	1.61
240	400.00	5,977	1.59
320	300.00	7,765	1.55
480	200.00	11,325	1.51
960	100.00	22,097	1.46
1,150	83.48	26,400	1.44
2,300	**41.74**	**52,225**	**1.40**
4,600	20.87	103,614	1.37
9,600	10.00	215,648	1.34
19,200	5.00	430,655	1.32
28,800	3.33	645,749	1.30
38,400	2.50	860,677	1.29
48,000	2.00	1,075,218	1.29
57,600	1.67	1,290,605	1.28
67,200	1.43	1,505,935	1.28
76,800	1.25	1,720,898	1.28
86,400	1.11	1,935,654	1.27
96,000	**1.00**	**2,150,010**	**1.27**

Table C.3 ***Influence of magnification factor on estimated fractal dimension fd_{LR} for the first 96×96 km^2 area.*** *With the increase of the magnification factor m the estimated fractal dimension fd_{LR} is decreasing. Convergence might be reached at fd_{LR} = 1.27 at m = 96,000.*

	interpolation method	interpolation length l_R	magnification factors m																							
			9,600	4,600	2,300	1,150	960	480	320	240	192	160	140	**120**	110	96	48	32	24	19	**16**	14	12	11	10	5
number points	original points		**13,761**	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	**13,761**	13,761	13,761	13,761	13,761	13,761
	doInterpm	**0.30**	**674,918**	327,414	167,478	87,505	74,217	40,890	29,808	24,296	20,900	18,532	17,113	15,852	15,321	14,703	13,836	13,778	13,767	13,763	**13,761**	13,761	13,761	13,761	13,761	13,761
		0.50	408,030	199,447	103,467	55,483	47,575	27,595	20,900	17,391	15,587	14,703	14,325	14,061	13,980	13,890	13,773	13,763	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.70	293,619	144,583	75,978	41,763	36,264	21,934	16,927	14,986	14,291	14,006	13,912	13,853	13,831	13,803	13,763	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.90	230,065	114,191	60,890	34,278	29,808	18,532	15,172	14,228	13,949	13,867	13,830	13,799	13,786	13,778	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.99	209,853	104,451	55,959	31,798	27,807	17,492	14,742	14,075	13,897	13,840	13,810	13,785	13,778	13,773	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
	doPythagoras	**0.30**	**759,020**	367,605	187,591	97,565	82,702	45,113	32,608	26,356	22,680	20,048	18,401	16,759	16,058	15,207	13,860	13,788	13,774	13,766	**13,761**	13,761	13,761	13,761	13,761	13,761
		0.50	458,484	223,585	115,607	61,608	52,626	30,141	22,680	18,711	16,413	15,207	14,615	14,244	14,103	13,959	13,781	13,766	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.70	329,616	161,881	84,680	46,114	39,780	23,735	18,182	15,579	14,544	14,160	13,997	13,888	13,856	13,822	13,767	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.90	258,031	127,621	67,590	37,622	32,608	20,048	15,806	14,472	14,057	13,908	13,850	13,818	13,801	13,788	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
		0.99	235,207	116,675	62,108	34,851	30,382	18,829	15,261	14,264	13,966	13,863	13,832	13,800	13,791	13,781	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761	13,761
number covered boxes	original points		**12,405**	12,386	12,246	11,780	11,518	9,416	7,327	5,820	4,743	3,966	3,484	2,948	2,741	2,330	1,021	601	404	280	215	173	130	111	97	25
	doInterpm	**0.30**	**263,989**	126,640	63,375	31,811	26,436	13,199	8,698	6,448	5,061	4,168	3,620	3,011	2,783	2,354	1,023	601	404	280	215	173	130	111	97	25
		0.50	246,182	118,352	59,553	29,910	25,004	12,585	8,391	6,237	4,923	4,059	3,542	2,984	2,760	2,341	1,023	601	404	280	215	173	130	111	97	25
		0.70	228,982	110,197	55,682	28,163	23,614	12,048	8,098	6,068	4,842	4,025	3,518	2,969	2,760	2,339	1,021	601	404	280	215	173	130	111	97	25
		0.90	211,652	102,377	52,110	26,670	22,407	11,553	7,853	5,999	4,806	4,009	3,513	2,964	2,755	2,335	1,021	601	404	280	215	173	130	111	97	25
		0.99	**203,752**	98,872	50,473	26,009	21,919	11,355	7,782	5,971	4,804	4,001	3,509	2,960	2,752	2,334	1,021	601	404	280	215	173	130	111	97	25
	doPythagoras	**0.30**	**267,920**	128,345	64,193	32,121	26,653	13,253	8,726	6,439	5,064	4,160	3,611	**3,008**	2,788	2,354	1,023	602	404	280	215	173	130	111	97	25
		0.50	253,097	121,024	60,716	30,442	25,320	12,634	8,386	6,215	4,910	4,050	3,548	2,979	2,762	2,342	1,023	601	404	280	215	173	130	111	97	25
		0.70	237,721	114,024	57,238	28,743	23,914	12,095	8,049	6,097	4,841	4,026	3,521	2,969	2,756	2,340	1,022	601	404	280	215	173	130	111	97	25
		0.90	222,454	106,801	53,768	27,014	22,695	11,573	7,835	6,001	4,816	4,005	3,508	2,964	2,754	2,338	1,021	601	404	280	215	173	130	111	97	25
		0.99	**215,648**	103,614	52,225	26,400	22,097	11,325	7,765	5,977	4,808	4,000	3,506	**2,961**	2,752	2,335	1,021	601	404	280	215	173	130	111	97	25
fd_{LR}	original points		**1.03**	1.12	1.22	1.33	1.36	1.48	1.54	1.58	1.61	1.63	1.65	1.67	1.68	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
	doInterpm	**0.30**	**1.36**	1.39	1.43	1.47	1.48	1.54	1.57	1.60	1.62	1.64	1.66	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.50	1.35	1.39	1.42	1.46	1.47	1.53	1.57	1.59	1.62	1.64	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.70	1.35	1.38	1.41	1.45	1.47	1.52	1.56	1.59	1.61	1.64	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.90	1.34	1.37	1.40	1.45	1.46	1.52	1.55	1.59	1.61	1.63	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.99	**1.33**	1.36	1.40	1.44	1.46	1.51	1.55	1.59	1.61	1.63	1.65	1.67	1.68	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
	doPythagoras	**0.30**	**1.36**	1.39	1.43	1.47	1.48	1.54	1.57	1.60	1.62	1.64	1.66	**1.67**	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.50	1.36	1.39	1.42	1.46	1.48	1.53	1.57	1.59	1.62	1.64	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.70	1.35	1.38	1.42	1.46	1.47	1.52	1.56	1.59	1.61	1.64	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.90	1.34	1.37	1.41	1.45	1.46	1.52	1.55	1.59	1.61	1.63	1.65	1.67	1.69	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00
		0.99	**1.34**	1.37	1.40	1.44	1.46	1.51	1.55	1.59	1.61	1.63	1.65	**1.67**	1.68	1.70	1.79	1.85	1.89	1.91	1.94	1.95	1.96	1.96	1.99	2.00

Table C.4 ***Influence of interpolation method and length on estimated fractal dimension fd_{LR} for the first 96×96 km² area.*** *With the decrease of the relative interpolation length l_R, the number of points and fd_{LR} are increasing.*

	square boxes		rectangular boxes		
m	***N***	fd_{LR}	***N***	fd_{LR}	**difference in** fd_{LR}
5	10	1.43	25	2.00	0.57
10	30	1.48	100	2.00	0.52
11	**33**	**1.46**	**121**	**2.00**	**0.54**
12	36	1.44	142	1.99	0.55
14	56	1.53	191	1.99	0.46
16	64	1.50	248	1.99	0.49
19	95	1.55	342	1.98	0.44
24	141	1.56	524	1.97	0.41
32	248	1.59	868	1.95	0.36
48	**520**	**1.62**	**1,704**	**1.92**	**0.31**
96	**1,652**	**1.62**	**4,604**	**1.85**	**0.22**
110	2,053	1.62	5,525	1.83	0.21
120	2,296	1.62	6,198	1.82	0.21
140	2,882	1.61	7,533	1.81	0.19
160	3,462	1.61	8,844	1.79	0.18
192	4,436	1.60	11,017	1.77	0.17
240	5,922	1.58	14,180	1.74	0.16
320	8,474	1.57	19,179	1.71	0.14
480	13,448	1.54	27,903	1.66	0.12
960	27,175	1.49	44,633	1.56	0.07
1,150	32,167	1.47	49,088	1.53	0.06
2,300	61,183	1.42	75,549	1.45	0.03
4,600	119,387	1.39	134,996	1.40	0.01
9,600	245,564	1.35	263,458	1.36	0.01
19,200	488,054	1.33	511,368	1.33	0.00
28,800	730,574	1.31	759,373	1.32	0.00
38,400	972,937	1.31	1,007,276	1.31	0.00
48,000	1,215,597	1.30	1,255,450	1.30	0.00
57,600	1,458,226	1.29	1,503,443	1.30	0.00
67,200	1,700,328	1.29	1,751,497	1.29	0.00
76,800	**1,943,066**	**1.29**	**1,999,432**	**1.29**	**0.00**
86,400	**2,185,183**	**1.28**	**2,247,502**	**1.29**	**0.00**
96,000	**2,428,097**	**1.28**	**2,495,701**	**1.28**	**0.00**

Table C.5 ***Influence of shape on estimated fractal dimension fd_{LR} for the complete 96 × 324 km^2 area.*** *Using rectangular boxes leads to a higher number of covered boxes N than using squares. This is because rectangular boxes are much smaller than square boxes. These range in the number of covered boxes transfers to the estimated fractal dimension. For both methods, the fd_{LR} converges at m = 96,000 to 1.28.*

	m_O	m_A	original position	horizontal translation step relative to box edge length h_r																				range
				0.000						0.125		0.250		0.375			0.500		0.625			0.750	0.875	
				vertical translation step relative to box edge length v_r																				
				0.250	0.375	0.500	0.625	0.750	0.875	0.125	0.250	0.125	0.625	0.125	0.375	0.875	0.125	0.375	0.000	0.125	0.250	0.625	0.375	
N	96	97	2,665	2,654	2,653	2,676	2,673	2,662	2,674	2,673	2,674	2,687	2,687	2,692	2,686	2,706	2,673	2,683	2,653	2,692	2,676	2,712	2,675	59
	48	49	1,147	1,151	1,147	1,148	1,148	1,142	1,144	1,163	1,169	1,171	1,167	1,156	1,167	1,162	1,163	1,171	1,162	1,160	1,156	1,164	1,179	37
	32	33	649	660	659	663	658	653	652	676	679	677	663	670	677	659	673	676	661	683	690	673	675	38
	24	**25**	422	436	426	427	421	**416**	418	444	**452**	446	430	446	437	440	444	434	431	443	449	440	438	**36**
	19	20	284	298	296	295	289	290	288	311	311	310	305	311	318	302	309	318	297	309	309	299	310	30
	16	17	215	215	218	219	221	220	218	235	226	237	233	237	229	237	234	228	229	235	226	233	226	22
	15	16	191	198	199	198	198	201	192	210	210	209	207	212	207	205	211	207	206	212	207	210	209	20
	14	15	172	178	171	169	173	173	173	188	190	188	185	187	182	184	192	186	185	193	194	184	181	25
	12	13	129	135	133	132	134	131	130	145	144	145	145	144	140	140	145	141	140	144	143	142	141	15
	11	**12**	107	114	117	114	114	111	**108**	123	124	123	123	123	125	117	121	125	116	125	**128**	124	126	**20**
	10	11	93	96	97	95	90	93	93	106	106	106	100	105	105	102	105	105	101	108	106	99	105	18
	5	**6**	25	**29**	28	28	28	26	**25**	**35**	**35**	34	34	34	34	30	34	34	30	34	34	32	34	**10**
fd_{LR}	96	**97**	1.73	1.72	1.72	1.73	1.72	1.72	1.72	1.72	1.72	0.00	1.73	1.73	1.73	1.73	1.73	1.72	1.73	1.72	1.73	1.73	1.73	1.73	**0.00**
	48	49	1.82	1.81	1.81	1.81	1.81	1.81	1.81	1.81	1.82	0.00	1.82	1.81	1.81	1.81	1.81	1.81	1.82	1.81	1.81	1.81	1.81	1.82	0.01
	32	33	1.87	1.86	1.86	1.86	1.86	1.85	1.85	1.86	1.86	0.00	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.87	1.87	1.86	1.86	0.02
	24	**25**	1.90	1.89	1.88	1.88	1.88	**1.87**	1.88	1.89	**1.90**	0.00	1.90	1.88	1.90	1.89	1.89	1.89	1.88	1.89	1.90	1.89	1.89	**0.03**
	19	20	1.92	1.90	1.90	1.90	1.89	1.89	1.89	1.92	1.92	0.00	1.91	1.91	1.92	1.92	1.91	1.91	1.92	1.90	1.91	1.91	1.90	1.91	0.03
	16	17	1.94	1.90	1.90	1.90	1.91	1.90	1.90	1.93	1.91	0.00	1.93	1.92	1.93	1.92	1.93	1.93	1.92	1.92	1.93	1.91	1.92	1.91	0.03
	15	16	1.94	1.91	1.91	1.91	1.91	1.91	1.90	1.93	1.93	0.00	1.93	1.92	1.93	1.92	1.92	1.93	1.92	1.92	1.93	1.92	1.93	1.93	0.04
	14	15	1.95	1.91	1.90	1.89	1.90	1.90	1.90	1.93	1.94	0.00	1.93	1.93	1.93	1.92	1.93	1.94	1.93	1.93	1.94	1.95	1.93	1.92	0.05
	12	13	1.96	1.91	1.91	1.90	1.91	1.90	1.90	1.94	1.94	0.00	1.94	1.94	1.94	1.93	1.93	1.94	1.93	1.93	1.94	1.93	1.93	1.93	0.04
	11	**12**	1.95	1.91	1.92	1.91	1.91	1.90	**1.88**	1.94	1.94	0.00	1.94	1.94	1.94	1.94	1.92	1.93	1.94	1.91	1.94	**1.95**	1.94	1.95	**0.07**
	10	11	1.97	1.90	1.91	1.90	1.88	1.89	1.89	1.94	1.94	0.00	1.94	1.92	1.94	1.94	1.93	1.94	1.94	1.92	1.95	1.94	1.92	1.94	0.08
	5	**6**	2.00	1.88	1.86	1.86	1.86	1.82	**1.80**	**1.98**	**1.98**	1.97	1.97	1.97	1.97	1.90	1.97	1.97	1.90	1.97	1.97	1.93	1.97	**0.19**

Table C.6 ***Influence of translation on estimated fractal dimension fd_{LR} for the fourth 96×96 km² area.*** *The original magnification factor m_O was adjusted to the new raster size as m_A. For the translated position of the vector data, the range in the fd_{LR} is up to 0.19. With the increase of m_A, the range in fd_{LR} is decreasing.*

	m_O (boxes in raster row for 0°)	box edge length	rotation angles																			range between all angles
			0	5	10	15	20	**25**	30	35	**40**	45	50	55	60	65	70	**75**	80	85	90	
N	96	$\frac{96,000}{96}$	3,524	3,576	3,587	3,624	3,599	3,607	3,617	3,588	3,621	3,617	3,602	3,597	3,627	3,593	3,613	3,585	3,567	3,533	3,524	103
	48	$\frac{96,000}{48}$	1,398	1,477	1,456	1,464	1,471	1,476	1,478	1,483	1,467	1,463	1,471	1,488	1,478	1,474	1,477	1,450	1,475	1,461	1,398	90
	32	$\frac{96,000}{32}$	775	815	806	796	791	799	822	815	819	817	809	823	825	826	818	803	798	797	775	51
	24	$\frac{96,000}{24}$	483	515	511	521	527	514	514	505	515	512	519	536	526	519	519	525	508	520	483	53
	19	$\frac{96,000}{19}$	323	351	352	371	371	342	353	363	357	353	360	365	352	365	359	357	358	351	323	48
	16	$\frac{96,000}{16}$	229	277	263	264	270	253	269	263	271	274	268	276	273	276	275	269	267	261	229	48
	15	$\frac{96,000}{15}$	209	242	239	246	233	233	250	244	236	233	244	244	242	241	238	246	239	234	209	41
	14	$\frac{96,000}{14}$	**185**	215	214	206	206	217	211	212	**216**	224	211	215	209	220	208	**226**	218	216	185	**41**
	12	$\frac{96,000}{12}$	136	155	163	167	167	161	171	166	167	163	164	166	175	159	167	161	165	162	136	39
	11	$\frac{96,000}{11}$	117	136	141	140	143	141	139	149	147	148	143	149	139	145	140	140	147	138	117	32
	10	$\frac{96,000}{10}$	**98**	116	124	123	125	**128**	121	118	**114**	115	119	117	121	126	119	124	125	116	98	**30**
	5	$\frac{96,000}{5}$	25	36	36	38	41	41	39	38	35	35	36	38	38	41	36	39	36	36	25	16
m_A	96	$\frac{96,000}{96}$	96	104	111	117	122	126	130	132	134	134	134	133	130	127	123	117	111	104	96	38
(maximum of boxes	48	$\frac{96,000}{48}$	48	52	56	59	61	63	65	66	67	67	67	67	65	64	62	59	56	52	48	19
in raster rows and columns)	32	$\frac{96,000}{32}$	32	35	37	39	41	42	44	44	45	45	45	45	44	43	41	39	37	35	32	13
	24	$\frac{96,000}{24}$	24	26	28	30	31	32	33	33	34	34	34	34	33	32	31	30	28	26	24	10
	19	$\frac{96,000}{19}$	19	21	22	24	25	25	26	27	27	27	27	27	26	26	25	24	22	21	19	8
	16	$\frac{96,000}{16}$	16	18	19	20	21	21	22	22	23	23	23	23	22	22	21	20	19	18	16	7
	15	$\frac{96,000}{15}$	15	17	18	19	19	20	21	21	21	21	21	21	21	20	20	19	18	17	15	6
	14	$\frac{96,000}{14}$	**14**	16	17	17	18	19	19	20	**20**	20	20	20	19	19	18	**18**	17	16	14	**6**
	12	$\frac{96,000}{12}$	12	13	14	15	16	16	17	17	17	17	17	17	17	16	16	15	14	13	12	5
	11	$\frac{96,000}{11}$	11	12	13	14	14	15	15	16	16	16	16	16	15	15	14	14	13	12	11	5
	10	$\frac{96,000}{10}$	**10**	11	12	13	13	**14**	14	14	**14**	14	14	14	14	14	13	13	12	11	10	**4**
	5	$\frac{96,000}{5}$	5	6	6	7	7	7	7	7	7	7	7	7	7	7	7	7	6	6	5	2
fd_{LR}	96	$\frac{96,000}{96}$	1.79	1.76	1.74	1.72	1.70	1.69	1.68	1.68	1.67	1.67	1.67	1.67	1.68	1.69	1.70	1.72	1.74	1.76	1.79	0.12
	48	$\frac{96,000}{48}$	1.87	1.85	1.81	1.79	1.77	1.76	1.75	1.74	1.73	1.73	1.73	1.74	1.75	1.75	1.77	1.79	1.81	1.84	1.87	0.14
	32	$\frac{96,000}{32}$	1.92	1.89	1.85	1.82	1.80	1.79	1.77	1.77	1.76	1.76	1.76	1.76	1.77	1.79	1.81	1.83	1.85	1.88	1.92	0.16
	24	$\frac{96,000}{24}$	1.94	1.92	1.87	1.84	1.83	1.80	1.79	1.78	1.77	1.77	1.77	1.78	1.79	1.80	1.82	1.84	1.87	1.92	1.94	0.18
	19	$\frac{96,000}{19}$	1.96	1.93	1.90	1.86	1.84	1.81	1.80	1.79	1.78	1.78	1.79	1.79	1.80	1.81	1.83	1.85	1.90	1.93	1.96	0.18
	16	$\frac{96,000}{16}$	1.96	1.95	1.89	1.86	1.84	1.82	1.81	1.80	1.79	1.79	1.78	1.79	1.81	1.82	1.84	1.87	1.90	1.93	1.96	0.18
	15	$\frac{96,000}{15}$	1.97	1.94	1.89	1.87	1.85	1.82	1.81	1.81	1.79	1.79	1.81	1.81	1.80	1.83	1.83	1.87	1.89	1.93	1.97	0.18
	14	$\frac{96,000}{14}$	**1.98**	1.94	1.89	1.88	1.84	1.83	1.82	1.79	**1.79**	1.81	1.79	1.79	1.81	1.83	1.85	**1.88**	1.90	1.94	1.98	**0.19**
	12	$\frac{96,000}{12}$	1.98	1.97	1.93	1.89	1.85	1.83	1.81	1.80	1.81	1.80	1.80	1.80	1.82	1.83	1.85	1.88	1.93	1.98	1.98	0.19
	11	$\frac{96,000}{11}$	1.99	1.98	1.93	1.87	1.88	1.83	1.82	1.80	1.80	1.80	1.79	1.80	1.82	1.84	1.87	1.87	1.95	1.98	1.99	0.20
	10	$\frac{96,000}{10}$	**1.99**	1.98	1.94	1.88	1.88	**1.84**	1.82	1.81	**1.79**	1.80	1.81	1.80	1.82	1.83	1.86	1.88	1.94	1.98	1.99	**0.20**
	5	$\frac{96,000}{5}$	2.00	2.00	2.00	1.87	1.91	1.91	1.88	1.87	1.83	1.83	1.84	1.87	1.87	1.91	1.84	1.88	2.00	2.00	2.00	0.17

Table C.7 ***Influence of rotation with square boxes on estimated fractal dimension fd_{LR} for the third 96×96 km² area .*** *The original magnification factor m_O was adjusted to the new raster size as m_A. Rotating vectors and using square boxes leads to a range in the estimated fractal dimension up to 0.20.*

	m_O	rotation angles																			range between all angles
		0	5	10	15	20	25	30	35	**40**	45	50	55	60	65	70	75	80	85	90	
N	**96**	**4,604**	3,806	3,247	2,850	2,572	2,389	2,266	2,164	**2,116**	2,082	2,093	2,129	2,209	2,337	2,491	2,784	3,201	3,786	4,604	**2,522**
	48	1,704	1,381	1,162	1,003	889	818	772	728	717	717	709	720	761	803	874	996	1,144	1,389	1,704	995
	32	868	706	588	506	450	406	390	366	357	357	361	368	384	411	450	506	582	719	868	511
	24	524	423	354	304	273	249	234	224	219	213	212	222	232	247	272	307	362	437	524	312
	19	**342**	280	237	200	179	164	157	148	**142**	141	145	149	154	163	182	204	236	284	342	**201**
	16	248	201	171	146	131	123	115	111	108	108	108	112	113	119	136	152	175	210	248	140
	15	220	180	151	134	120	107	99	99	96	94	97	98	103	110	118	133	152	184	220	126
	14	191	158	132	117	105	95	88	88	86	82	82	89	91	98	105	116	133	163	191	109
	12	142	117	102	90	79	76	68	67	66	68	67	66	68	73	80	90	100	121	142	76
	11	121	101	84	73	69	65	63	59	57	57	58	58	59	64	71	76	86	103	121	64
	10	**100**	84	72	65	58	55	49	49	**47**	49	49	52	53	55	59	64	74	87	100	**53**
	5	25	23	21	19	18	17	15	15	15	15	15	17	17	17	18	18	21	24	25	10
fd_{LR}	**96**	**1.85**	1.81	1.77	1.74	1.72	1.70	1.69	1.68	**1.68**	1.67	1.68	1.68	1.69	1.70	1.71	1.74	1.77	1.81	1.85	**0.17**
	48	1.92	1.87	1.82	1.79	1.75	1.73	1.72	1.70	1.70	1.70	1.70	1.70	1.71	1.73	1.75	1.78	1.82	1.87	1.92	0.23
	32	1.95	1.89	1.84	1.80	1.76	1.73	1.72	1.70	1.70	1.70	1.70	1.70	1.72	1.74	1.76	1.80	1.84	1.90	1.95	0.26
	24	1.97	1.90	1.85	1.80	1.77	1.74	1.72	1.70	1.70	1.69	1.69	1.70	1.71	1.73	1.76	1.80	1.85	1.91	1.97	0.28
	19	**1.98**	1.91	1.86	1.80	1.76	1.73	1.72	1.70	**1.68**	1.68	1.69	1.70	1.71	1.73	1.77	1.81	1.86	1.92	1.98	**0.30**
	16	1.99	1.91	1.85	1.80	1.76	1.74	1.71	1.70	1.69	1.69	1.69	1.70	1.71	1.72	1.77	1.81	1.86	1.93	1.99	0.30
	15	1.99	1.92	1.85	1.81	1.77	1.73	1.70	1.70	1.69	1.68	1.69	1.69	1.71	1.74	1.76	1.81	1.86	1.93	1.99	0.31
	14	1.99	1.92	1.85	1.80	1.76	1.73	1.70	1.70	1.69	1.67	1.67	1.70	1.71	1.74	1.76	1.80	1.85	1.93	1.99	0.32
	12	1.99	1.92	1.86	1.81	1.76	1.74	1.70	1.69	1.69	1.70	1.69	1.69	1.70	1.73	1.76	1.81	1.85	1.93	1.99	0.31
	11	2.00	1.92	1.85	1.79	1.77	1.74	1.73	1.70	1.69	1.69	1.69	1.69	1.70	1.73	1.78	1.81	1.86	1.93	2.00	0.31
	10	**2.00**	1.92	1.86	1.81	1.76	1.74	1.69	1.69	**1.67**	1.69	1.69	1.72	1.72	1.74	1.77	1.81	1.87	1.94	2.00	**0.33**
	5	2.00	1.95	1.89	1.83	1.80	1.76	1.68	1.68	1.68	1.68	1.68	1.76	1.76	1.76	1.80	1.80	1.89	1.97	2.00	0.32

Table C.8 ***Influence of rotation with rectangular boxes on estimated fractal dimension fd_{LR} for the complete 96 × 324 km^2 area.*** *The rotation of the vectors for the 96×384 km^2 area leads to a range in the fd_{LR} up to 0.33. The range in the estimated fractal dimension is ≥ 0.17 at all magnifications factors m_O.*

	magnification factor	rotation angles																			range between all angles
		0	5	10	15	20	25	30	35	**40**	**45**	50	55	60	65	70	75	80	85	90	
N	**96**	2,699	2,705	2,724	2,733	2,723	2,704	2,732	2,730	2,727	2,742	2,742	2,725	2,735	2,697	2,717	2,680	2,733	2,695	2,699	**62**
	48	1,115	1,121	1,123	1,118	1,119	1,122	1,119	1,120	1,119	1,134	1,146	1,125	1,117	1,128	1,112	1,112	1,122	1,123	1,115	34
	32	633	625	620	625	634	630	627	627	635	626	633	630	625	636	631	620	640	621	633	20
	24	406	398	400	398	390	397	407	395	399	396	404	399	401	396	395	399	398	406	406	17
	19	275	272	270	273	275	267	274	273	271	271	267	277	275	270	268	273	272	273	275	10
	16	196	204	206	202	204	201	202	202	201	204	197	205	204	203	208	198	205	207	196	12
	14	164	163	161	164	158	163	158	159	159	160	163	163	162	161	159	158	162	163	164	6
	12	124	124	125	122	123	124	125	122	124	121	123	126	124	125	121	128	125	128	124	7
	11	106	109	103	108	105	102	107	105	104	104	107	106	107	106	105	106	107	104	106	7
	10	87	87	87	87	85	87	86	87	85	87	86	85	87	86	87	88	86	86	87	3
	5	25	25	25	25	25	25	25	25	24	24	25	25	25	25	25	25	25	25	25	**1**
fd_{LR}	**96**	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	1.73	**0.01**
	48	1.81	1.81	1.81	1.81	1.81	1.81	1.81	1.81	1.81	1.82	1.82	1.81	1.81	1.82	1.81	1.81	1.81	1.81	1.81	0.01
	32	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	1.86	0.01
	24	1.89	1.88	1.89	1.88	1.88	1.88	1.89	1.88	1.88	1.88	1.89	1.88	1.89	1.88	1.88	1.88	1.88	1.89	1.89	0.01
	19	1.91	1.90	1.90	1.91	1.91	1.90	1.91	1.91	1.90	1.90	1.90	1.91	1.91	1.90	1.90	1.91	1.90	1.91	1.91	0.01
	16	1.90	1.92	1.92	1.91	1.92	1.91	1.91	1.91	1.91	1.92	1.91	1.92	1.92	1.92	1.93	1.91	1.92	1.92	1.90	0.02
	14	1.93	1.93	1.93	1.93	1.92	1.93	1.92	1.92	1.92	1.92	1.93	1.93	1.93	1.93	1.92	1.92	1.93	1.93	1.93	0.01
	12	1.94	1.94	1.94	1.93	1.94	1.94	1.94	1.93	1.94	1.93	1.94	1.95	1.94	1.94	1.93	1.95	1.94	1.95	1.94	0.02
	11	1.94	1.96	1.93	1.95	1.94	1.93	1.95	1.94	1.94	1.94	1.95	1.94	1.95	1.94	1.94	1.94	1.95	1.94	1.94	0.03
	10	1.94	1.94	1.94	1.94	1.93	1.94	1.93	1.94	1.93	1.94	1.93	1.93	1.94	1.93	1.94	1.94	1.93	1.93	1.94	0.02
	5	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	**1.97**	**1.97**	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	**0.03**

Table C.9 ***Influence of rotation circles on estimated fractal dimension fd_{LR} for the third 96×96 km² area.*** *The rotation of circular areas has less effect on the estimated fractal dimension than the rotation of a square areas. The maximum range is only 0.03 at m = 5 and decreases to 0.01 m = 19.*

m	box edge length in metre	N	fd_{FD}	fd_{MD}	fd_{LR}	difference fd_{MD} and fd_{LR}
5	19,200.00	25		1.97	2.00	0.03
10	9,600.00	98	1.97	1.96	1.99	0.03
11	8,727.27	117	1.86	1.80	1.99	0.19
12	8,000.00	136	1.73	1.90	1.98	0.08
14	6,857.14	185	2.00	1.81	1.98	0.17
16	6,000.00	**229**	1.60	**1.81**	1.96	0.15
19	5,052.63	**322**	1.98	**1.84**	1.96	0.12
24	4,000.00	**483**	1.74	**1.68**	1.94	0.26
32	3,000.00	775	1.64	1.54	1.92	0.38
48	2,000.00	1,400	1.46	1.38	1.87	0.49
96	1,000.00	3,516	1.33	1.30	1.79	0.48
110	872.73	4,131	1.18	1.21	1.77	0.56
120	800.00	4,604	1.25	1.18	1.76	0.58
140	685.71	5,489	1.14	1.10	1.74	0.64
160	600.00	6,326	1.06	1.04	1.72	0.68
192	500.00	7,628	1.03	1.02	1.70	0.68
240	400.00	9,567	1.02	0.98	1.67	0.69
320	300.00	**12,570**	**0.95**	0.92	1.64	**0.72**
480	200.00	**18,074**	**0.90**	0.93	1.59	0.66
960	100.00	34,753	**0.94**	0.95	1.52	0.57
1,150	83.48	41,544	0.99	0.98	1.51	0.53
2,300	41.74	81,732	0.98	0.98	1.46	0.48
4,600	20.87	162,146	0.99	0.99	1.42	0.43
9,600	10.00	336,789	0.99	1.00	1.39	0.39
19,200	5.00	672,427	1.00	1.00	1.36	0.36
28,800	3.33	1,007,998	1.00	1.00	1.35	0.35
38,400	2.50	1,343,331	1.00	1.00	1.34	0.34
48,000	2.00	1,679,029	1.00	1.00	1.33	0.33
57,600	1.67	2,014,999	1.00	1.00	1.32	0.32
67,200	1.43	2,350,462	1.00	1.00	1.32	0.32
76,800	1.25	2,685,577	1.00	1.00	1.32	0.32
86,400	1.11	3,021,474	1.00	1.00	1.31	0.31
96,000	1.00	3,356,992	1.00	1.00	1.31	0.31

fd_{GS}	$fd_{GM(FD)}$	$fd_{GM(MD)}$	$fd_{GM(LR)}$
1.10	**1.24**	**1.25**	**1.64**

Table C.10 ***Influence of different calculation methods on estimated local and global fractal dimension for the first 96×96 km² area.*** *With increase of the magnification factor m, fd_{LR} continuously decreases, but fd_{FD} and fd_{MD} change the direction. The differences between the estimated local fractal dimension transfers to the estimated global fractal dimensions.*

	figure	pattern form	area number	shape	original magnification factor **5**	adjusted magnification factor **5**	cell edge length	interpolation method	interpolation length l_R	multiple matrices or single matrix	position	N	fd_{LR}	range in fd_{LR}
a)	m = 9,600 too high for figure	96×96 km^2	1	squares	9600	9600	$\frac{1}{9600}$	**original points** **doInterpm** **doPythagoras**	0.3	multiple	original	**12,405** **263,989** **267,920**	**1.03** **1.36** **1.36**	**0.33**
b)	m = 9,600 too high for figure	96×96 km^2	1	squares	9600	9600	$\frac{1}{9600}$	doPythagoras	**0.30** **0.99**	multiple	original	**267,920** **215,648**	**1.36** **1.34**	**0.02**
c)	figure 5.5 figure 5.6	96×96 km^2	1	squares	**96,000** **2,300** **140** **5**	**96,000** **2,300** **140** **5**	$\frac{1}{96000}$ $\frac{1}{2300}$ $\frac{1}{140}$ $\frac{1}{5}$	doPythagoras	0.99	multiple	original	**2,150,010** **52,225** **3,506** **25**	**1.27** **1.40** **1.65** **2.00**	**0.73**
d)	figure 5.8	96×324 km^2		**squares** **rectangles**	11	11	$\frac{1}{11}$	doPythagoras	0.99	multiple	original	**33** **121**	**1.46** **2.00**	**0.54**
e)	figure 5.9	96×96 km^2	4	squares	11	**12**	$\frac{1}{11}$	doPythagoras	0.99	multiple	**translation** ($h_r = \frac{7}{8}$, $v_r = \frac{0}{8}$) **translation** ($h_r = \frac{5}{8}$, $v_r = \frac{2}{8}$)	**108** **128**	**1.88** **1.95**	**0.07**
f)	figure 5.10	96×96 km^2	3	squares	10	**10** **14**	$\frac{1}{10}$	doPythagoras	0.99	multiple	**rotation (0°)** **rotation (40°)**	**98** **114**	**1.99** **1.79**	**0.20**
g)	figure 5.10	96×324 km^2		rectangles	24	24	$\frac{1}{24}$	doPythagoras	0.99	multiple	**rotation (0°)** **rotation (50°)**	**524** **212**	**1.97** **1.69**	**0.28**
h)	figure 5.10	96×324 km^2		rectangles	10	10	$\frac{1}{10}$	doPythagoras	0.99	multiple	**rotation (0°)** **rotation (40°)**	**100** **47**	**2.00** **1.67**	**0.33**
i)	figure 5.10	circular with radius 48km	3	squares	5	5	$\frac{1}{5}$	doPythagoras	0.99	multiple	**rotation (0°)** **rotation (40°)**	**25** **24**	**2.00** **1.97**	**0.03**

Table C.11 ***Summary Table for Fault Lines.*** *There is a big range in the Box-Counting fractal dimension based on different methods. One of the highest differences with 0.54 is for the shape of the box, rectangular versus square boxes. The rotation of the vectors lead to a difference up to 0.33. Different magnification factors result in a range of 0.73. The column multiple or single matrices is kept for the comparison with table C.2*

Appendix D

Matlab Codes

Even for the appendix, the complete functions would have been too long, thus I only present the most important excerpts.

Code D.1 *read shapefile and change organization*

```
[ result ] = function read_shapefile_and_change_element_by_element_2_plane ( dataset )
element_organization = shaperead (dataset); % read shapefile (.shp)
fields = fieldnames (element_organization) ;
for f = 1:size(fields,1); % change element-by-element to plane_organization
varname = genvarname(fields{f}) ; % curly bracket as fields is cell
extract                     = {element_organization.(varname)}; % dynamic field
plane_organization.(varname)     = extract ; % dynamic field
% save result
end
```

Code D.2 *rotating based on original GIS data*

```
function [ result ] = rotating_based_on_original_GIS_data ( data , angle_dataset )
[Y_Ori, X_Ori ] = polyjoin(data.Y, data.X) ;
angle = -angle_dataset % here angle_dataset = 36.16
    X_R = X_Ori * cosd(angle) - Y_Ori * sind(angle) ;
    Y_R = X_Ori * sind(angle) + Y_Ori * cosd(angle) ;
% save as 'result'
end
```

Code D.3 *Box-Counting Fault Lines*

```

% Note, the functions D4 - D10 are subfunctions of D3

function [xyz  ] =  Box_Counting_Fault_Lines ( dataset )
%% user-made choices
result.distance_calculations  =   { 'doPythagoras' } ; % {'original_points'}  {'doInterpm'}
result.interpolation_lengths_relative = [ 0.99   ] ; % this is l_R
result.multiple_matrices =     {'yes'} ; % {'no'} % this is multiple or single matrix
result.position_changes =    {'rotation'} ; %  {'translation'} {'constant'}
result.circles         =     {'yes' } ; %  {'no'} % this is for circular analysis area
% this is how the matrix representing the raster is increased
result.pattern_forms =  {'original_size'} ; % {'original_size' 'square' 'square2n' }
```

```
13 result.cell_forms =   {'squares'} ;% {'rectangles'} % this is box shape
14 %this is for indicating if rectangular areas are compared with square areas
15 % so that the a rectangular area of analysis is chosen
16 result.compare_squares_rectangles =     {'no' } ; %  {'yes'} %
17 result.magnification_factors =   [ 96000  5 ] ; % here are all m
18 result.cell_edge_lengths_relative = [ 1./result.magnification_factors ];
19 % in the thesis I only showed the areas of 96000*96000 square meters (or the multiple 96*324 km^2
20 result.all_cut_areas =   [   96000 ];% [ 96000 48000 32000 ]
21 for ca = 1 : size(result.all_cut_areas,2) ;
22     cut_area = result.all_cut_areas(1,ca) ;
23     cut_area_size(1) = { [ 'cut_area_' num2str(cut_area) ] } % put fieldname together
24     varname_cut_area = genvarname( cut_area_size{1}); % creates a variable with the name cut_area_96000
25     square_in_row = 0 ; % this is the position for the result in the table
26     square_in_col = 0 ; % this is the position for the result in the table
27     %% get fault line data - this data are already rotated with 36.16 degree
28     [ ~ , X_R , Y_R , ~ , dataset , ~ ] = get_data ( dataset, result ) ;
29     if size(X_R,1) == 1 % for areas smaller than 96*96km^2 there are sometimes no fault lines
30         if X_R == 'NaN'
31             error(['X_columnvector_rotated is empty' ]);
32         end
33     end
34     minXLimWorld = min(X_R) ; %% determine limits of data
35     maxXLimWorld = max(X_R) ; %% determine limits of data
36     minYLimWorld = min(Y_R) ; %% determine limits of data
37     maxYLimWorld = max(Y_R) ; %% determine limits of data
38     %% data extraction - determine north and south of cut area
39     for row_start_square_in_Y         = 0     : cut_area : (maxYLimWorld-minYLimWorld) - cut_area ;
40         square_in_row = square_in_row + 1 ;
41         square_in_col = 0 ;
42         Y_N_for_trimming = maxYLimWorld  - (square_in_row-1)*cut_area ;
43         Y_S_for_trimming = Y_N_for_trimming -                   cut_area  ;
44         %% data extraction - determine west and east of cut area
45         cols_X = 0     : cut_area : (maxXLimWorld-minXLimWorld) - cut_area ;
46         for col_start_square_in_X =  0     : cut_area : (maxXLimWorld-minXLimWorld) - cut_area ;
47             square_in_col = square_in_col + 1 ;
48             westSquareX = minXLimWorld + (square_in_col-1)*cut_area ;
49             eastSquareX  = westSquareX +                   cut_area  ;
50         %% if you need rectangular area, keep west border of first 96*96 area
51         if     (strcmp(cell_form ,'rectangles') &&  square_in_col == 1) || ...
52                (strcmp(cell_form ,'squares') && strcmp(result.compare_squares_rectangles ,'yes' ) && ...
53                square_in_col == 1) ;
54                keep_westSquareX_from_first_loop = westSquareX ;
55         elseif (strcmp(cell_form ,'rectangles') &&  square_in_col ~= 1) || ...
56                (strcmp(cell_form ,'squares') && strcmp(result.compare_squares_rectangles ,'yes' ) &&
...
57                square_in_col ~= 1) ;
58                westSquareX = keep_westSquareX_from_first_loop ;
59         end
60             Xlim_trimmed = [ westSquareX eastSquareX ] ;
61             Ylim_trimmed = [ Y_S_for_trimming Y_N_for_trimming ] ;
62             %% start with main loops for user-made choices
63             for cf = 1      :size(result.cell_forms,2)
64                 cell_form = result.cell_forms(cf) ;
65                 varname_cell_form = genvarname( char(cell_form) ) ;
66                 subplot_number = 0
67                 for cel = 1      :size(result.cell_edge_lengths_relative,2)
68                     e_R = result.cell_edge_lengths_relative(1,cel)  ;
69                     mf      = result.magnification_factors(1,cel)  ;
70                     varname_mf = genvarname( [ 'magnification_factor_' num2str(mf) ] ) ;
```

```
                    longer_edge = max( abs(diff(Xlim_trimmed)) , abs(diff(Ylim_trimmed)) ) ;
                    e_U = longer_edge / mf ; % absolute cell edge length
                    for ptr = 1      : size(result.position_changes,2)
                        position_change = result.position_changes(ptr) ;
                        varname_position = genvarname( [ char(position_change) ] ) ;
                        for  dc = 1      : size(result.distance_calculations,2)
                            int_method = result.distance_calculations(dc) ;
                            varname_distance_calculation = genvarname( [ char(int_method) ] ) ;
                            for mm =  1 : size(result.multiple_matrices,2)
                                mmYesNo = result.multiple_matrices(mm) ;
                                varname_mmYesNo = genvarname( [ char(mmYesNo) ] ) ;
                                for pf = 1      : size(result.pattern_forms,2)
                                    pattern_form = result.pattern_forms(pf) ;
                                    varname_pattern_form = genvarname( [ char(pattern_form) ] ) ;
                                    for ir = 1:size(result.interpolation_lengths_relative,2)
                                        l_R = result.interpolation_lengths_relative(ir) ;
                                        varname_inter_pol = genvarname( char(l_R) ) ;
    %% determine steps and number of loops for translation and rotation
    if strcmp ( position_change , 'translation')
        moving_cell_size = 1  ; % total move in cell edge lengths
        step_moves = 8 % number of steps for total move
        moving_from_step_to_step = moving_cell_size / step_moves  ; % move per step
        % following order is important for correct subplot plotting
        all_step_h_R = [ 0 : moving_from_step_to_step : moving_from_step_to_step*(step_moves-1)
] ;
        all_step_v_R =  [ moving_from_step_to_step*(step_moves-1)  : - moving_from_step_to_step : 0
] ;
        biggest = length(all_stepsizes_X_absolute) ;  % important for plotting
        all_steps =  [ biggest     1:biggest-1  biggest+1: length(all_stepsizes_X_absolute)^2
] ;
    elseif strcmp ( position_change ,'rotation')
        all_angles =  [ 0:5:90 ] ; % angles from 0 to 90 in steps of 5
        if strcmp ( cell_form , 'squares' )
            biggest = 10
        elseif strcmp ( cell_form , 'rectangles' )
            biggest = 10
        end
        all_steps =  [ biggest     1 : biggest-1  biggest+1: length(all_angles)   ] ;
    elseif strcmp ( position_change , 'constant') ==1
        disp (['constant' ]) ;
        all_steps =  1 ;
    else error(['Invalid optional argument, ' char(position_change)  ]);
    end

    %% defining plots for translation, rotations, and constant positions
    % removed in thesis appendix as not result relevant

%% go through all steps
for step = all_steps

    %% trim fault lines to square or rectangular area
    [  Y_T , X_T        ] = ...
        trimPolylineToQuadrangle (  Y_R , X_R ,  Ylim_trimmed , Xlim_trimmed  ) ;
    clear ('X_columnvector_rotated' , 'Y_columnvector_rotated' );
    %% trim fault lines to circular area
    if  strcmp ( result.circles , 'yes' )
        [  X_T , Y_T , radius ,  westSquareX , eastSquareX , southSquareY , northSquareY ] = ...
            trimPolylineToCircle_myown  ( X_T  , Y_T  , Xlim_trimmed , Ylim_trimmed  ) ;
    end
```

```
127
128     % here was some saving of result
129
130     %% determine amount of position change
131     if  strcmp ( position_change , 'translation') %% determine horizontal and vertical step
132         step_Y = ceil ( step / length(all_stepsizes_X_absolute))   ;
133         step_X = step - (step_Y-1)*length(all_stepsizes_X_absolute)    ;
134         stepsize_X_relative = all_step_h_R(step_X) ;
135         stepsize_Y_relative = all_step_v_R(step_Y) ;
136         h_U = all_stepsizes_X_absolute(step_X) ;
137         v_U = all_stepsizes_Y_absolute(step_Y) ;
138         % translate vector
139         [ translation ] = comparing_translation ( X_T , Y_T ,  e_U , h_U , v_U  );
140         varname_translation_or_rotation = genvarname( [ char(fieldnames(translation)) ] ) ;
141         X_Y_TraRotNone.X = translation.(varname_translation_or_rotation).X_translated ;
142         X_Y_TraRotNone.Y = translation.(varname_translation_or_rotation).Y_translated ;
143         % here was some saving of result
144     elseif strcmp ( position_change ,'rotation') %% determine angle for rotation step
145         angle = all_angles(step) ;
146         Koch_Curve = 'no'; % only important for input in next function
147         [ rotation , radius ] = comparing_rotation_grid_Koch_Curve_newest ...
148                                 ( X_T , Y_T , cell_form , angle , Koch_Curve ) ;
149         varname_translation_or_rotation = genvarname( [ char(fieldnames(rotation)) ] ) ;
150         X_Y_TraRotNone.X = rotation.(varname_translation_or_rotation).X_rotated ;
151         X_Y_TraRotNone.Y = rotation.(varname_translation_or_rotation).Y_rotated ;
152         clear ( 'rotation' )
153     else %% this is for constant position
154         X_Y_TraRotNone.X = X_T  ;
155         X_Y_TraRotNone.Y = Y_T ;
156     end
157
158 %% create Matlab mapraster reference object
159 if strcmp ( position_change , 'translation')
160   [R_Z]  = maprasterref_Z_empty_newest     ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y , ...
161   e_R , e_U , cell_form , westSquareX , eastSquareX , Y_S_for_trimming , Y_N_for_trimming
    , ...
162   'translation' , 'yes' , 'pattern_form' , pattern_form ) ;
163 elseif strcmp ( position_change ,'rotation') && strcmp ( result.circles , 'no' )
164   [R_Z]  = maprasterref_Z_empty_newest    ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y , ...
165   e_R , e_U , cell_form , westSquareX , eastSquareX , Y_S_for_trimming , Y_N_for_trimming  ,
    ...
166   'rotation' , 'yes' , 'radius' , radius , 'pattern_form' , pattern_form , 'circles' , result.circles );
167 elseif strcmp ( position_change ,'rotation')  && strcmp ( result.circles , 'yes' )
168   [R_Z]  = maprasterref_Z_empty_newest    ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y , ...
169   e_R , e_U , cell_form , westSquareX , eastSquareX , southSquareY , northSquareY  ,  ...
170   'rotation' , 'yes' , 'radius' , radius , 'pattern_form' , pattern_form , 'circles' , result.circles );
171 else
172   [R_Z ] = maprasterref_Z_empty_newest     ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y , ...
173   e_R , e_U , cell_form , westSquareX , eastSquareX , Y_S_for_trimming , Y_N_for_trimming  )
    ;
174 end
175
176     R       = R_Z.R ;
177     Z_empty = R_Z.Z_empty ;
178
179     %% interpolation for multiple matrices
180     if strcmp ( cell_form , 'squares' ) ==1 % different absolute interpolation length
181         if  strcmp ( int_method , 'doInterpm' ) || strcmp ( int_method , 'doPythagoras' )
182             l_U = e_U * l_R ; % maximum distance is the length of the diagonal of the rectangular box
```

```
183             [X_L , Y_L ] = doInterpolation_Fault_Lines ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y
    , l_U , 'distance_calculation' , int_method ) ;
184         elseif strcmp ( int_method , 'original_points' )
185             if  strcmp ( position_change , 'translation')   || strcmp ( position_change ,'rotation')
186                 X_L = X_Y_TraRotNone.X ;
187                 Y_L = X_Y_TraRotNone.Y ;
188             end
189         end
190     elseif strcmp ( cell_form , 'rectangles' ) == 1 % different absolute interpolation length
191         if  strcmp ( int_method , 'doInterpm' ) || strcmp ( int_method , 'doPythagoras' )
192             e_U_height = abs(R.DeltaY) ; % cell edge height
193             e_U_width  = abs(R.DeltaX) ; % cell edge width
194             % instead of cell edge length, I took the length of the diagonal of the rectangular box
195             l_U = sqrt(e_U_height^2+e_U_width^2) * l_R ;
196             [X_L , Y_L ] = doInterpolation_Fault_Lines ( X_Y_TraRotNone.X , X_Y_TraRotNone.Y
    , ...
197                 l_U , 'distance_calculation' , int_method) ;
198         elseif  strcmp ( int_method , 'original_points' )
199             if strcmp ( position_change , 'translation')   || strcmp ( position_change ,'rotation')
200                 X_L = X_Y_TraRotNone.X ;
201                 Y_L = X_Y_TraRotNone.Y ;
202             end
203         end
204     end
205
206     %% here, results are saved in a Matlab structure
207
208     %% convert vector into raster data
209     [col, row] = setpostn(Z_empty, R, X_L, Y_L) ;
210     %% create matrix reflecting the raster data, fill empty grid cells
211     % (based in Matlab build-in function 'imbedm'
212     value = 1
213     value = value + zeros(size(row)) ;
214     qNaN = isnan(col) | isnan(row) ;
215     col  (qNaN) = [] ;
216     row  (qNaN) = [] ;
217     value(qNaN) = [] ;
218     index = (col-1)*size(Z_empty,1) + row ;  %%% linear matrix indexing
219
220     Z_empty(index) = value ;
221     Z_binary = Z_empty ;
222     adapted_magnification_factor = max ( size(Z_binary,1) , size(Z_binary,2) ) ;
223     % adapt magnification factor to higher number of rows or columns
224     Z_binary = sparse (Z_binary) ; %% the sparse form needs less computational power
225
226     fd_LR_    = log(sum(sum(Z_binary))) ./ log(result.magnification_factors(cel)) ;
227     fd_LR_    = (round(fd_LR_*100))/100 ;
228
229     adapted_fd_LR    = log(sum(sum(Z_binary))) ./ log(adapted_magnification_factor) ;
230     adapted_fd_LR    = (round(adapted_fd_LR*100))/100 ;
231
232     %% here, results are saved in a Matlab structure
233
234     %% tables I save earlier
235     % table_Z_binary_sum is the count -> (sum(sum(Z_binary)))
236     % table_adapted_magnification_factor is the maximum number of rows or columns of raster
237     %                                         -> max ( size(Z_binary,1) , size(Z_binary,2) )
238     % table_magnification_factor is a list of all magnification factors
239
```

```
% calculate different local fractal dimensions
fd_FD  = +diff(log(table_Z_binary_sum))    ./diff(log(table_magnification_factor));
fd_MD  = +gradient(log(table_Z_binary_sum)).gradient(log(table_magnification_factor)) ;
fd_LR  = log(table_Z_binary_sum)           ./log(table_magnification_factor) ;
fd_LR_adapted_magnification  = log(table_Z_binary_sum) ./ log(table_adapted_magnification_factor) ;

% calculate different global fractal dimensions based on mean
fd_GM_FD = mean(fd_FD(isfinite(fd_FD))) ;  % isfinite is false for nan as well as -inf and +inf
fd_GM_MD = mean(fd_MD(isfinite(fd_MD))) ; % isfinite is false for nan as well as -inf and +inf
fd_GM_LR = mean(fd_LR(isfinite(fd_LR))) ; % isfinite is false for nan as well as -inf and +inf

% calculate different global fractal dimensions based on slope
% via polyfit % results in two numbers (slope and intercept)
fd_GS_slope_polyfit = polyfit (log(table_magnification_factor), log(table_Z_binary_sum) ,1)
;
fd_GS_polyfit_slope           = fd_GS_slope_polyfit(1) ;
% via regression % results in regression values R, slope, and y-intercept
[ R , fd_GS_slope , y_intercept ] = regression(log(table_magnification_factor), log(table_Z_binary_sum)) ;
fd_GS_regression_slope        = slope ;

    %% all these result were saved in a Matlab structure and finally saved as output 'xyz'

                                    end
                                end
                            end
                        end
                    end
                end
            end
        end
    end
end
end
end
```

Code D.4 *trim Polyline to Quadrangle*

```
% The original Matlab function is called maptriml.
% It is for trimming geographic coordinates.
% However, the fault line data are in projected map data and
% did not fulfill requirements of sub-function of maptriml, such as trimPolylineToQuadrangle.
function [ xnew , ynew ] = trimPolylineToQuadrangle_newest ( x , y , xlim , ylim )

% For all four borders we can use the same trim function.
% For 'lower' the matrix gets reflected and then the same procedure can be applied as for 'upper'

% Trim latitudes to southern limit.
  [y, x] = trimPolylineToVerticalLine(y, x, ylim(1), 'lower');

% Trim latitudes to northern limit.
  [y, x] = trimPolylineToVerticalLine (y, x, ylim(2), 'upper');

% Trim unwrapped longitudes to the western limit.
  [x, y] = trimPolylineToVerticalLine(x, y, xlim(1), 'lower');

% Now trim to the eastern limit.
  [x, y] = trimPolylineToVerticalLine(x, y, xlim(2), 'upper');

```

```
xnew = x ;
ynew = y ;

end

%---------------------------------------------------------------------

function [x, y] = trimPolylineToVerticalLine (x, y, xBound, boundType)
% sign flip to reflect across x == 0, lower bounds can be treated the same way as upper bounds.
% Then reflect across y == 0 so to preserved right-handedness.
usingLowerBound = strncmpi(boundType, 'lower', numel(boundType));
if usingLowerBound ;
    x = -x;
    y = -y;
    xBound = -xBound;
end

% interpolating an extra point each time a curve crosses the vertical line x == xBound,
% eliminating bounds % where x > xBound.
% The last argument (false) indicates that we want to trim lines instead of polygons.
[x, y] = truncateAtBoundary (x, y, xBound, false);
% 'truncateAtBoundary' is an original Matlab function and therefore not further presented

% Reverse any sign flips.
if usingLowerBound ;
    x = -x;
    y = -y;
end
```

Code D.5 *trim Polyline To Circle*

```
function [ X_T , Y_T , radius , X_W_centered , X_E_centered , Y_S_centered , northSquareY_C ] = ...
         trimPolylineToCircle  ( X_T , Y_T , Xlim_trimmed , Ylim_trimmed )

westSquareX  = Xlim_trimmed(1) ;
eastSquareX  = Xlim_trimmed(2) ;
southSquareY = Ylim_trimmed(1) ;
northSquareY = Ylim_trimmed(2) ;

%% center the square area to the origin
X_C_original = x - mean ( [ eastSquareX westSquareX ] ) ;
X_E_centered         = eastSquareX - mean ( [ eastSquareX westSquareX ] ) ;
X_W_centered         = westSquareX - mean ( [ eastSquareX westSquareX ] ) ;

Y_C_original = y - mean ( [ northSquareY southSquareY ] ) ;
Y_N_centered         = northSquareY - mean ( [ northSquareY southSquareY ] ) ;
Y_S_centered         = southSquareY - mean ( [ northSquareY southSquareY ] ) ;

X_C = X_C_original ;
Y_C = Y_C_original ;

radius1 = (X_E_centered  - X_W_centered) /2 ;
radius2 = (Y_N_centered - Y_S_centered) /2 ;

if radius1 ~= radius2
    disp ( [  westSquareX , eastSquareX , southSquareY , northSquareY ] )
    error('myApp:argChk', 'difference -> (eastSquareX-westSquareX) /2 ~=
(northSquareY-southSquareY) /2')
```

```
end

radius = radius1 ;

%% check which point are inside or outside of circle
distance_point_from_center = sqrt (X_C.^2 + Y_C.^2 ) ;
signDiff = diff(sign(radius - distance_point_from_center)) ;
kCrossing = find(abs(signDiff) == 2); % find crossing of line through circle
% results only in 2 if one of the points is inside and one outside, if both outside it is zero.
kCrossing = flipud(kCrossing); % because there will be new rows added

for j = 1:numel(kCrossing)
    k = kCrossing(j)  ;

    %% check direction of lines: from inside to outside OR outside to inside
    crossingInsideToOutside = (radius - distance_point_from_center(k)) >= 0  ;
    % 1 for yes -> crossing Inside to Outside
    % 0 for no  -> crossing Outside to Inside

    p = polyfit( [X_C(k); X_C(k+1) ] , [Y_C(k) ; Y_C(k+1) ] , 1 ) ;
    slope       = p(1) ;
    intercept   = p(2) ;
    if isnan(slope(1))
        break
    else
        [xout ,yout] = linecirc(slope,intercept, 0 , 0 ,radius);
        % in-built Matlab function that finds the points of intersection given a circle
        % defined by a center and radius in x-y coordinates,
        % and a line defined by slope and y-intercept
    end
    % I get each 2 points for x and y (because a line with slope and y-intercept
    (but no start and end points crosses twice the circle
    % now I need to check which of these two points fall between my original points

        for col = 1:size(xout,2) % same check as above but for each point separate
            if (    min(X_C(k),X_C(k+1)) < xout(1,col) && xout(1,col) < max(X_C(k),X_C(k+1)) && ...
                    % check if new 'xout' is between original x coordinate values
                    min(Y_C(k),Y_C(k+1)) < yout(1,col) && yout(1,col) < max(Y_C(k),Y_C(k+1)) && ...
                    % check if new 'yout' is between original y coordinate values

                x_on_circle = xout(1,col) ;
                y_on_circle = yout(1,col) ;

        [X_C, Y_C] = insertForLine(k, x_on_circle , y_on_circle, X_C, Y_C , crossingInsideToOutside);
                % I adapted the Matlab built-in function to circles
            end
        end

end

% Remove points beyond the boundary. This is the last step AFTER all points were inserted
[  x_clipped, y_clipped] = truncateForLine(X_C, Y_C , radius ) ;
end

%-------------------------------------------------------------------------

% I adjusted the Matlab built-in 'insertForLine' and 'truncateForLine' to circles
% (the original version is a subfunction of the function maptriml
% the path is maptriml -> trimPolylineToQuadrangle ->
```

```
% trimPolylineToVerticalLine -> truncateAtBoundary -> insertForLine and truncateForLine
% (maptriml trims a line with vertices specified by vectors to the quadrangle)

function [X_C, Y_C] = insertForLine(k, x_on_circle, y_on_circle, X_C , Y_C , crossingInsideToOutside );
if crossingInsideToOutside
    xInsert = [x_on_circle; NaN];
    yInsert = [y_on_circle; NaN];
else
    xInsert = [NaN; x_on_circle];
    yInsert = [NaN; y_on_circle];
end
X_C = [X_C(1:k); xInsert; X_C((k+1):end)];
Y_C = [Y_C(1:k); yInsert; Y_C((k+1):end)];
end

%--------------------------------------------------------------------------

function [x_clipped , y_clipped ] = truncateForLine(X_C , Y_C  , radius ) ;
x_clipped = X_C ;
y_clipped = Y_C ;
q = (x_clipped.^2 + y_clipped.^2 > radius^2 ) ;
x_clipped (q) = NaN;
y_clipped(q) = NaN;
q = isnan(x_clipped);
q = q & [q(2:end); false];
x_clipped(q) = [];
y_clipped(q) = [];
end
```

Code D.6 *comparing translation*

```
function [ translation ] = comparing_translation ( X , Y , e_U , h_U , v_U  )

minXLimWorld = min(X) ;
maxXLimWorld = max(X) ;
minYLimWorld = min(Y) ;
maxYLimWorld = max(Y) ;

% adding_X =  cell_edge_length_absolute * stepsize_X_relative ;
% X_translated = X + adding_X ;
X_translated = X + h_U ;

% adding_Y =  cell_edge_length_absolute * stepsize_Y_relative ;
% Y_translated = Y + adding_Y ;
Y_translated = Y + v_U ;

varname_adding_X_Y = genvarname( [ 'stepsize_X_'  num2str(h_U)  '_stepsize_Y_'
num2str(v_U) ]) ;
translation.(varname_adding_X_Y).adding_X     = h_U ;
translation.(varname_adding_X_Y).adding_Y     = v_U ;
translation.(varname_adding_X_Y).cell_edge_length_absolute = e_U ;
translation.(varname_adding_X_Y).X_translated = X_translated ;
translation.(varname_adding_X_Y).Y_translated = Y_translated ;

end
```

Code D.7 *comparing rotation*

```
1
2  function [ rotation , radius ] = comparing_rotation ( X_C , Y_C , cell_form, angle ,
   Koch_Curve )
3  westSquareX   = min(X) ;
4  eastSquareX   = max(X) ;
5  southSquareY  = min(Y) ;
6  northSquareY  = max(Y) ;
7
8  radius = (eastSquareX-westSquareX) /2 ;
9
10 angle = angle
11 angle_size(1) = { [ 'angle_' num2str(angle) ] } ;
12 varname_angle = genvarname( angle_size{1}) ;
13
14 X_rotated = X_C * cosd(angle) - Y_C * sind(angle) ;
15 Y_rotated = X_C * sind(angle) + Y_C * cosd(angle) ;
16
17 rotation.(varname_angle).X_rotated = X_rotated ;
18 rotation.(varname_angle).Y_rotated = Y_rotated ;
19 end
```

Code D.8 *maprasterref and empty grid*

```
1  function [ R_Z] = maprasterref_Z_empty ( X , Y  , e_R , e_U , cell_form , ...
2      westSquareX , eastSquareX , southSquareY , northSquareY ,   varargin )
3
4  if strcmp (cell_form , 'squares')  == 1
5      if exist('translation','var') == 1
6  Xlim = [ westSquareX     (round(max(X(:))*1000))/1000] ; % keep west as minimum
7  Ylim = [ southSquareY     max(Y(:)) ]; % keep south as minimum so that grid starts there
8  n_C = ceil(diff(Xlim)/e_U) ;  % cells in columns
9  n_R = ceil(diff(Ylim)/e_U) ;  % cells in rows
10 Xlim = [ westSquareX   westSquareX+n_C*e_U]; % new Xlim max to make squared boxes
11 Ylim = [ southSquareY  southSquareY+n_R*e_U]; % new Ylim max to make squared boxes
12     elseif exist('rotation','var') == 1 && strcmp ( circles , 'yes' ) == 0
13 Xlim = [ min(X)        max(X(:)) ] ;
14 Ylim = [ min(Y)        max(Y(:)) ];
15 n_C = ceil(diff(Xlim)/e_U);
16 n_R = ceil(diff(Ylim)/e_U);
17     elseif exist('rotation','var') == 1 && strcmp ( circles , 'yes' ) == 1
18 Xlim = [ westSquareX       eastSquareX  ] ;
19 Ylim = [ southSquareY      northSquareY ];
20 n_C = ceil(diff(Xlim)/e_U);
21 n_R = ceil(diff(Ylim)/e_U);
22     else
23 Xlim = [ min(X) (round(max(X(:))*1000)/1000)] ;
24 Ylim = [ min(Y)        max(Y(:)) ];
25 n_C = ceil(diff(Xlim)/e_U);
26 n_R = ceil(diff(Ylim)/e_U);
27 Xlim = [min(X)  min(X)+n_C*e_U]; % new Xlim max to make squared boxes
28 Ylim = [min(Y)  min(Y)+n_R*e_U]; % new Ylim max to make squared boxes
29     end
30 end
31
32 if strcmp (cell_form , 'rectangles')  == 1
33     if exist('translation','var') == 1
```

```
xy_min = 0 ; % for translation keep 0 as minimum, otherwise use min(Y(:))
Xlim = [ xy_min        max(X(:)) ] ;
Ylim = [ xy_min        max(Y(:)) ];
n_C = 1/e_R;  % equals magnification factor
n_R = 1/e_R;
    elseif exist('rotation','var') == 1
Xlim = [ xy_min        max(X(:)) ] ;
Ylim = [min(Y(:))       max(Y(:))];
n_C = 1/e_R;  % equals magnification factor
n_R = 1/e_R;
    else
Xlim = [ xy_min        max(X(:)) ] ;
Ylim = [min(Y(:))       max(Y(:))];
n_C = 1/e_R;  % equals magnification factor
n_R = 1/e_R;
    end
end

if     exist('pattern_form','var') == 1 && strcmp ( pattern_form , 'square' )
    n_R = max( n_C , n_R )
    n_C = max( n_C , n_R )
    width = max( n_C , n_R )
    Xlim = [Xlim(1)  Xlim(1)+n_C*e_R]; % new Xlim max to make squared boxes
    Ylim = [Ylim(1)  Ylim(1)+n_R*e_R]; % new Ylim max to make squared boxes
elseif exist('pattern_form','var') == 1 && strcmp ( pattern_form , 'square2n' ) % such as for Moisy
    width = max( n_C , n_R );
    power = log(width)/log(2);   % nbre of generations
    if (power~=round(power) || any(size(c)~=width) ) && strcmp(pattern,'Koch_Snowflake')
            power = ceil(power);
            n_C = 2^power;
            n_R = 2^power;
    elseif (power~=round(power) || any(size(c)~=width) ) &&  strcmp(pattern,'Koch_Curve')
            power = ceil(power);
            n_C = 2^power;
            n_R = 2^power;
    end
        Xlim = [Xlim(1)  Xlim(1)+n_C*e_R]; % new Xlim max to make squared boxes
    Ylim = [Ylim(1)  Ylim(1)+n_R*e_R]; % new Ylim max to make squared boxes
end

sizeZ =   ceil([n_R n_C ])  ;  % size of raster
Z_empty = zeros(sizeZ); % empty matrix reflecting the size of the raster

% create mapraste reference object
R = maprasterref;
R.XLimWorld  = Xlim;
R.YLimWorld  = Ylim;
R.RasterSize = sizeZ;
R.RowsStartFrom    = 'west' ;
R.ColumnsStartFrom = 'south' ;
R_Z.R       = R ;
R_Z.Z_empty = Z_empty ;

end
```

Code D.9 *Interpolation of Fault Lines*

```
function [X , Y ] = Interpolation_of_Fault_Lines ( X , Y , l_U , varargin )
```

```
%% access_variables_and_values_from_varargin
    if (~isempty(varargin))
        for c=1:2:length(varargin)
             switch varargin{c}
                case {'distance_calculation'}
                    int_method = varargin{c+1}  ; % the right to variable
                otherwise
                error(['Invalid optional argument, ' varargin{c}]);
            end % switch
        end % for
    end % if

X = X(:) ;
Y = Y(:) ;

if strcmp ( int_method , 'doInterpm' ) == 1
dist  = max( [abs(diff(X))'; abs(diff(Y))'] )'; %
indx  = find( dist > l_U) ;
elseif strcmp ( int_method , 'doPythagoras' ) == 1
dist  = sqrt ( abs(diff(X)).^2 + abs(diff(Y)).^2 )' ; % this is
indx  = find( dist > l_U) ;
end

format longG  ; nansum(dist) ;

if ~isempty(indx) ;
     steps = ceil(dist(indx)/(l_U)) ;      %  No points added each location
     totalpts = sum(steps)-length(steps) ;   %  Total points to be added
     lastpt  = length(X) ;                  %  Current last point in data set
     X(length(X)+totalpts) = 0 ;         %  Pre-allocate output memory
     Y(length(Y)+totalpts) = 0 ;
end

for i=length(indx):-1:1 ;
    if strcmp ( int_method , 'doInterpm' ) == 1

        loc = indx(i) ; %  set index in the original vectors and compute the interpolation steps.
        factors = (1:steps(i)-1)' ; %  -1 eliminates double hit at end of interpolation insert

        % evenly space interpolation points
        latinsert = ((X(loc+1)-X(loc))/steps(i))*factors + X(loc) ;
        loninsert = ((Y(loc+1)-Y(loc))/steps(i))*factors + Y(loc) ;


        X=[X(1:loc); latinsert; X(loc+1:lastpt)] ; %  Fill in the interpolated data
        Y=[Y(1:loc); loninsert; Y(loc+1:lastpt)] ; %  Fill in the interpolated data
        %  Update the last point of the data set.  Note that since
        %  the output memory is pre-allocated, the current last point
        %  of the data set is not equal to the length of the data vector
        lastpt = lastpt + length(latinsert) ;
    elseif strcmp ( int_method , 'doPythagoras' ) == 1
        loc = indx(i) ;
        factors = (1:steps(i)-1)' ; %  -1 eliminates double hit at end of interpolation insert

        % consecutively  added points
        latinsert = (X(loc+1)-X(loc)) / (dist(loc)/(l_U)) *factors + X(loc) ;
        loninsert = (Y(loc+1)-Y(loc)) / (dist(loc)/(l_U)) *factors + Y(loc) ;

        %  Fill in the interpolated data
```

```
        X=[X(1:loc); latinsert; X(loc+1:lastpt)] ;
        Y=[Y(1:loc); loninsert; Y(loc+1:lastpt)] ;
        %  Update the last point of the data set.  Note that since
        %  the output memory is pre-allocated, the current last point
        %  of the data set is not equal to the length of the data vector
        lastpt = lastpt + length(latinsert) ;
    end
end
end
```

Code D.10 *Convert Vector to Raster Coordinates*

```
function [col, row , col_unrounded , row_unrounded ] = convert_vector_to_raster_coordinates
(Z, R, X_T, Y_T)
[col_unrounded, row_unrounded] = R.worldToIntrinsic(X_T, Y_T) ;
col = round(col_unrounded) ;
row = round(row_unrounded) ;
end
```

Code D.11 *Compare distribution of the orientation*

```

% Note, the function D12 is a subfunction of D11

function [lmn] =  Compare_distribution_of_the_orientation_of_fault_lines ( dataset )

%% user choices

result.all_cut_areas =  [ 48000 ]; % [ 96000  48000 32000 ] % size of area, e.g. 48x48 km^2
result.comparison_weight =  'unweighted'; % 'weighted';% weighted with lengths
comparison_w    = { [ 'comparison_weight_' result.comparison_weight ] } ; % put fieldname together
varname_comparison_weight = genvarname( comparison_w {1,1} )    ;

result.number_permutations = 999  ;

result.only_some_areas_of_interest =          'yes';%         'no';%
result.areas_of_interest = [ 2,2 ; 2,4 ; 2,8 ] ; % e.g, are 11 (9+2),13 (9+4) and 17 (9+8)
result.which_KS_test =       '2KS';%      'MKS';%

result.bin_size = ( 1 ) ;
bin_size{1}   = { [ 'bin_size' num2str(result.bin_size) ] } ; % put fieldname together
varname_bin_size = genvarname( bin_size{1}) ;

for ca =  size(result.all_cut_areas,2) ;
    data = importdata  ( sprintf('S:/...) ;

    if strcmp (result.only_some_areas_of_interest , 'no' ) % means all areas are interesting
        % determine the number of areas, save as 'entries'
    end

    % determine number of loops for testing
    if strcmp ( result.which_KS_test , '2KS' )
        loops = entries^2;
    elseif strcmp ( result.which_KS_test , 'MKS' )
        loops = 1;
    end

```

```
    for loop = 1:loops

        %% get radians for areas of interest - not presented in thesis because of extent
        % result is two matlab structures called 'angles' and 'input_length'

        for i = 1 :     length(fields(angles)) % two for 2KS, more than 2 for MKS
        var_x = genvarname ( [ 'x' num2str(i) ] ) ;

        %% rad2deg and keep between 0 and 180
        angles.(var_x) = rad2deg (angles.(var_x))    ;
        angles.(var_x)(angles.(var_x)>=180) = angles.(var_x)(angles.(var_x)>=180)-180 ;
        angles.(var_x)(angles.(var_x)<0)    = angles.(var_x)(angles.(var_x)<0)+180 ;

        %% delete NaN and ceil length
        input_length.(var_x)  =  input_length.(var_x)(~isnan(input_length.(var_x)));
        input_length.(var_x)  =  input_length.(var_x)(:);
        input_length.(var_x)  = (ceil (input_length.(var_x)/100)) ;
        % ceil lengths of fault lines to 100 meter for weighting
        end

    %% weight or unweighted
    if strcmp (varname_comparison_weight , 'comparison_weight_unweighted' )
        clear input_length
        if length(fields(angles)) == 2 % do 2KS test
        [ h_decision , p_value ,ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 (angles) ;
        result.original_data.h_decision     (position.square_1 , position.square_2) = h_decision ;
        result.original_data.p_value        (position.square_1 , position.square_2) = p_value ;
        result.original_data.ks2statistics  (position.square_1 , position.square_2) = ks2statistics ;
        elseif length(fields(angles)) > 2 % do MKS test
        [ ~ , ~ , ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 ( angles) ;
        result.original_data.ks2statistics(1,1) = ks2statistics ;
        end
    elseif  strcmp (varname_comparison_weight , 'comparison_weight_weighted' )
        for i = 1 : length(fields(angles)) % replicate angles
            var_x = genvarname ( [ 'x' num2str(i) ] ) ;
            length_original_AND_new_vectors.(var_x) = length (input_length.(var_x)) ;
            % keep size for repmat so that n1,n2 for calculating critical values are old size
            repmat_big = [] ;
            for j = 1 : length (angles.(var_x)) ;
                % replicate each angle with the number of the weight
                repmat_small = repmat ( angles.(var_x)(j) , input_length.(var_x)(j) , 1) ;
                row_repmat_small = size ( repmat_small ,1 )   ;
                row_repmat_big   = size ( repmat_big ,1 ) ;
                % put replicated angles together
                repmat_big ( row_repmat_big+1:row_repmat_big+row_repmat_small , 1 ) = repmat_small ;
                weighted_angles.(var_x) = repmat_big ;
            end
        end
        if length(fields(angles)) == 2  % do 2KS test
            [ h_decision , p_value ,ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 ...
                ( weighted_angles  , 'size' , length_original_AND_new_vectors ) ;
        result.original_data.h_decision   (position.square_1 , position.square_2) = h_decision ;
        result.original_data.p_value      (position.square_1 , position.square_2) = p_value ;
        result.original_data.ks2statistics(position.square_1 , position.square_2) = ks2statistics ;
        elseif length(fields(angles)) > 2 % do MKS test
            [ ~ , ~ , ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 ...
                ( weighted_angles  , 'size' , length_original_AND_new_vectors ) ;
        result.original_data.ks2statistics(position.square_1 , position.square_2) = ks2statistics
;
```

```
95          end
96      else
97          error(['varname_comparison_weight not a valid input'  ])
98      end
99
100     %% permutation of the whole groups without replacement and randomization
101
102     permutations.h_decision     = []   ;
103     permutations.p_value        = []   ;
104     permutations.ks2statistics  = []   ;
105     clear index_group_real ;
106     size_groups = [] ;
107     for i = 1 : length(fields(angles))
108         var_x = genvarname ( [ 'x' num2str(i) ] ) ;
109         % make vector for group labels
110         angles.(var_x)  =  angles.(var_x)(~isnan(angles.(var_x)));
111         angles.(var_x)  =  angles.(var_x)(:);
112         row_needed = size ( angles.(var_x) ,1 )   ;
113         size_groups   = size ( index_group_real ,1 ) ;
114         index_group_real ( size_groups+1:size_groups+row_needed , 1 ) = i ;
115         angles_together ( size_groups+1:size_groups+row_needed , 1 ) = angles.(var_x) ;
116         if  strcmp (varname_comparison_weight , 'comparison_weight_weighted' )
117             length_together ( size_groups+1:size_groups+row_needed , 1 ) = input_length.(var_x) ;
118         end
119     end
120
121     index_group_permutation_all = []; % need empty matrix before loop
122     permutation_angles = []   ; % need empty matrix before loop
123     permutation_length = []   ; % need empty matrix before loop
124
125     %% do permutations, in most cases I did 999 permutations
126     p = 1   ;
127     while p <= result.number_permutations
128
129         %% permutation of group labels
130
131         index_group_permutation   = index_group_real(randperm(size(index_group_real,1)),:)
;
132         index_group_permutation_all  ( : , p ) = index_group_permutation   ;
133
134         %% is same permutation already in use? compare all permutations
135         if p >= 2
136             for pp = 1:p-1
137                 if index_group_permutation_all ( : , pp ) == index_group_permutation
138                     is_it_alredy_existing = 'yes'   ; % don't use it again
139                     break
140                 else
141                     is_it_alredy_existing = 'no'   ;
142                 end
143             end
144             if strcmp ( is_it_alredy_existing , 'yes' )
145                 continue
146             end
147         end
148
149         %% get new groups
150         for i = 1 : length(fields(angles))
151             var_x = genvarname ( [ 'x' num2str(i) ] ) ;
152             indices  =  find ( index_group_permutation == i )   ;
```

```
153             permutation_angles.(var_x)  = angles_together (indices)   ;
154
155             %% check size - something must be totally wrong
156             if size ( angles.(var_x),1)  ~=  size ( permutation_angles.(var_x),1) ;
157                 error(['something must be totally wrong with indices'  ])
158             end
159
160             if  strcmp (varname_comparison_weight , 'comparison_weight_weighted' )
161                 permutation_length.(var_x)  = length_together (indices)   ;
162             end
163
164         end
165
166         %% compare permutation_angles of areas
167
168         if strcmp (varname_comparison_weight , 'comparison_weight_unweighted' )
169             if length(fields(permutation_angles)) == 2
170             [ h_decision , p_value ,ks2statistics , ~ , ~ , ~  ] = ...
171                 multiple_kstest2 ( permutation_angles) ;
172             permutations.h_decision     ( 1 , p )= h_decision    ;
173             permutations.p_value        ( 1 , p )= p_value   ;
174             permutations.ks2statistics  ( 1 , p )= ks2statistics    ;
175             % h = 0 => Do not reject the null hypothesis at the 5% significance level.
176             % h = 1 => Reject the null hypothesis at the 5% significance level.
177             elseif length(fields(permutation_angles)) > 2
178             [ ~ , ~ ,ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 ( permutation_angles) ;
179             permutations.ks2statistics  ( 1 , p )= ks2statistics    ;
180             end
181         elseif  strcmp (varname_comparison_weight , 'comparison_weight_weighted' )
182             for i = 1 : length(fields(permutation_angles))
183             var_x = genvarname ( [ 'x' num2str(i) ] ) ;
184             repmat_big = [] ; % start empty matrix otherwise elements get added !
185             for j = 1 : length (permutation_angles.(var_x)) ;
186             repmat_small = repmat ( permutation_angles.(var_x)(j) , permutation_length.(var_x)(j) , 1) ;
187             row_repmat_small = size ( repmat_small ,1 )   ;
188             row_repmat_big   = size ( repmat_big ,1 ) ;
189             repmat_big ( row_repmat_big+1:row_repmat_big+row_repmat_small , 1 ) = repmat_small ;
190             end
191             weighted_angles.(var_x) = repmat_big ;
192             end
193
194             if length(fields(angles)) == 2 % 2KS test
195             [ h_decision , p_value ,ks2statistics , ~ , ~ , ~  ] = multiple_kstest2 ...
196                 ( weighted_angles , 'size' , length_original_AND_new_vectors ) ;
197             permutations.h_decision     ( 1 , p ) = h_decision    ;
198             permutations.p_value        ( 1 , p ) = p_value   ;
199             permutations.ks2statistics  ( 1 , p ) = ks2statistics    ;
200             % h = 0 => Do not reject the null hypothesis at the 5% significance level.
201             % h = 1 => Reject the null hypothesis at the 5% significance level.
202             elseif length(fields(angles)) > 2 % MKS test
203             [ ~ , ~ ,ks2statistics , ~ , ~ , ~ ] = multiple_kstest2 ...
204                 ( weighted_angles , 'size' , length_original_AND_new_vectors ) ;
205             permutations.ks2statistics  ( 1 , p )= ks2statistics    ;
206             end
207         else
208             error(['varname_comparison_weight not a valid angles'  ])
209         end
210
211         %% comparison_original_with_permutation.p_value
```

```
212
213         [ ~ , ~ , index ] = find ( permutations.ks2statistics (permutations.ks2statistics >= ...
214                     result.original_data.ks2statistics(position.square_1 , position.square_2))) ;
215         size_index = length ( index )  ;
216         comparison_original_with_permutation.last_p_value  ( 1 , p ) = size_index / p
    ; % p-value after each permutation
217
218         p = p + 1 ;
219         if rem ( p , 100 ) == 0
220             disp([' p: ' num2str(p) ]) ;
221         end
222     end
223         % save result as 'lmn' structure
224     end
225 end
226 end
```

Code D.12 *MKS and 2KS test*

```
1  function [ H, pValue, KSstatistic , ~ , ~ , ~ ] = MKS_and_2KS_test ( input , varargin )
2
3  %% access_variables_and_values_from_varargin
4  if (~isempty(varargin))
5      for c=1:2:length(varargin)
6          switch varargin{c}
7              case {'alpha_niveau'}
8                  alpha_niveau = varargin{c+1} ;  % the right to variable
9              case {'size'}
10                 size = varargin{c+1} ;  % the right to variable
11             otherwise
12                 error(['Invalid optional argument, ' varargin{c}]);
13         end % switch
14     end % for
15 end % if
16
17 %% calculate binedges
18 input_binedges = [ ] ;
19
20 for i = 1 : length(fields(input))
21     var_x = genvarname ( [ 'x' num2str(i) ] ) ;
22     input.(var_x)  =  input.(var_x)(~isnan(input.(var_x)));
23     input.(var_x)  =  input.(var_x)(:);
24
25     row_needed = length ( input.(var_x))   ;
26     row_input_binedges   = length ( input_binedges ) ;
27     input_binedges ( row_input_binedges+1:row_input_binedges+row_needed , 1 ) = input.(var_x) ;
28 end
29
30 %% Calculate F1(x) and F2(x), the empirical (i.e., sample) CDFs.
31
32 binEdges    =  [ 0 ; sort(input_binedges) ; 180 ];
33
34 for i = 1 : length(fields(input))
35     var_x = genvarname ( [ 'x' num2str(i) ] ) ;
36     varname_sampleCDF = genvarname ( [ 'sampleCDF' num2str(i) ] ) ;
37
38     binCounts  =  histc (input.(var_x) , binEdges, 1);
39     sumCounts  =  cumsum(binCounts)./sum(binCounts); % Cumulative sum of elements.
```

```
    sampleCDF  =  sumCounts(1:end-1);

    sampleCDFs.(varname_sampleCDF) = sampleCDF ;
    sampleCDFs_all(:,i) =  sampleCDF ;
end

deltaCDF  =  abs((max(sampleCDFs_all,[],2)) - (min(sampleCDFs_all,[],2)));
KSstatistic   =  max(deltaCDF);


%% This code is from the Matlab built-in function kstest2 for the Two-sample Kolmogorov-Smirnov test
if length(fields(input)) == 2
    n1     =  length(input.x1);
    n2     =  length(input.x2);
    if exist ( 'size' , 'var' ) % for the weighting version the original number of the angles is used
                                % rather than the number of replicated angles
        n1 = size.x1 ;
        n2 = size.x2 ;
    end
    n      =  n1 * n2 /(n1 + n2);
    lambda =  max((sqrt(n) + 0.12 + 0.11/sqrt(n)) * KSstatistic , 0);
    %% 2-sided test (default).
    %  Use the asymptotic Q-function to approximate the 2-sided P-value.
    j       =  (1:101)';
    pValue  =  2 * sum((-1).^(j-1).*exp(-2*lambda*lambda*j.^2));
    pValue  =  min(max(pValue, 0), 1);
    if exist ( 'alpha_niveau' , 'var')
        if ~isscalar(alpha_niveau) || (alpha_niveau <= 0 || alpha_niveau >= 1)
            error(message('stats:kstest2:BadAlpha'));
        end
    else
        alpha_niveau  =  0.05;
    end
    H  =  (alpha_niveau >= pValue);
else
    pValue = [] ;
    H = [] ;
end
end
```

List of Figures

List of Tables

References

[1] Fractal Antennas - Front Page. http://www.tsc.upc.es/fractalcoms/fractals/gallery/spk60.html, accessed 24 Oct 2013.

[2] Home | Life Hackery - Part 9. http://lifehackery.com/category/blogs/home/page/9/, accessed 22 Oct 2013.

[3] Is the Cloud an Energy Catastrophe? | Cloud Storm Chasers. http://blogs.ca.com/cloud/2012/09/27/is-the-cloud-an-energy-catastrophe/, accessed 18 Apr 2013.

[4] Khajuraho Temples, India - Sightseeing in India photos. http://www.worldisround.com/articles/369798/photo129.html, accessed 20 Oct 2013.

[5] "Koru - fern frond" by Jenny Dean | Redbubble. http://www.redbubble.com/people/jennydean/works/10027527-koru-fern-frond, accessed 22 Oct 2013.

[6] Lake Wanaka, Mountains, New Zealand | Top travel lists. http://toptravellists.net/lake-wanaka-mountains-new-zealand.html, accessed 05 May 2013.

[7] Modelování v 3DS Max 3 - Kaktus - Grafika.cz - vše o počítačové grafice. http://www.grafika.cz/rubriky/3dscena/modelovani-v-3ds-max-3-kaktus-131135cz, accessed 19 Oct 2013.

[8] Page 1 - Spirals - The Easy Way step-by-step tutorial. http://www.xaraxone.com/webxealot/workbook22/page_1.htm, accessed 23 Oct 2013.

[9] Pont du Gard roman bridge - The Pont du Gard roman bridge - 04 - Landscape-Photo.net. http://www.landscape-photo.net/displayimage.php?pid=2365, accessed 19 Oct 2013.

[10] Styropor-Kugeln-1-Teilig.jpg. http://www.kreativnihobi.com/files/Image/VIVA_DECOR_NOVI/Styropor-Kugeln-1-Teilig.jpg, accessed 24 Apr 2013.

[11] SWR Warum? Glühlampe / Gasfüllung. http://www.planet-schule.de/warum/gluehlampe/themenseiten/t6/s1.html, accessed 24 Oct 2013.

[12] The Measure of Happiness: How YOU Feel | emotionistic. http://emotionistic.com/posts/the-measure-of-happiness-how-you-feel/, accessed 23 Oct 2013.

[13] Why the Fern? Fern Life Center. http://www.fernlifecenter.com/about-fern-life/why-the-fern/, accessed 13 Oct 2013.

[14] Fractal Antenna for Digital TV, 2012. http://www.cqdx.ru/ham/do-it-yourself/diy-fractal-antenna-for-digital-tv/, accessed 24 Oct 2013.

[15] Anoop.

[16] R. Aratsu. JYI : Leonardo Was Wise - Trees Conserve Cross-Sectional Area Despite Vessel Structure. *Journal of Young Investigators*, 1(1), 1998.

[17] Y. Ben-Zion and C. G. Sammis. Characterization of fault zones. *Pure and Applied Geophysics*, 160, 2003.

[18] B. A. Bradley and M. Hughes. Spatial distribution of ground motion. Technical report, University of Canterbury, 2012.

[19] H. Campbell. 2010 Darfield Earthquake, 2010. http://www.scientists.org.nz/discuss/2010/10/2010-darfield-earthquake, accessed 27 Apr 2013.

[20] E. H. Christiansen. Earth's Dynamic Systems Web Edition 1.0, 2009. http://www.earthds.info/, accessed 27 Apr 2013.

[21] College of the Redwoods. mantle convection. http://dn.redwoods.edu/coursenotes/renner/geo_images/plate_tectonics/mantle_convection.jpg, accessed 28 Apr 2013.

[22] A. Costa. hausDim.m - Hausdorff (Box-Counting) Fractal Dimension, 2011. http://www.mathworks.com/matlabcentral/fileexchange/30329-hausdorff-box-counting-fractal-dimension/content/hausDim.m, accessed 19 Apr 2012.

[23] P. A. Cowie, C. Vanneste, and D. Sornette. Statistical Physics Model for the Spatiotemporal Evolution of Faults. *Journal of Geophysical Research*, 98(B12):21809–21821, 1993.

[24] A. Crivoi and F. Duan. Evaporation-induced formation of fractal-like structures from nanofluids. *Physical chemistry chemical physics : PCCP*, 14(4):1449–54, 2012.

[25] Culturally Situated Design Tools. African Fractals, 2013. http://csdt.rpi.edu/african/african_fractals/applications5.html, accessed 25 Sep 2013.

[26] A. E. Egger. Plate Tectonics II: Plates; Plate Boundaries; and Driving Forces, 2003. http://www.visionlearning.com/library/module_viewer.php?mid=66, accessed 28 Apr 2013.

[27] R. Eglash, O., and Toluwalogo B. Fractals, Complexity, and Connectivity in Africa. In *What Mathematics from Africa?* Polimetrica International Scientific Publisher Monza/Italy, 2005.

[28] C. Eloy. Leonardo's rule, self-similarity and wind-induced stresses in trees. *Physical Review Letters*, 107:258101, 2011.

[29] Encyclopædia Britannica. Gondwana. http://www.britannica.com/EBchecked/topic/238402/Gondwana, accessed 08 Oct 2013.

[30] Encyclopædia Britannica. plate tectonics: crustal generation and destruction, 2007. http://kids.britannica.com/comptons/art-107937, accessed 12 Oct 2013.

[31] H. Fossen. *Structural Geology*. Cambridge University Press, 2010.

[32] Fractal Explorer. Fractals In Our World. http://www.wahl.org/fe/Flash_version/introduction.swf, accessed 08 Sep 2013.

[33] P. French. Fractal dimension measurement.m - Hausdorff Dimension by the box counting method, 2007. http://www.mathworks.com/matlabcentral/fileexchange/15918-hausdorff-dimension-by-the-box-counting-method/content/Fractaldimensionmeasurement.m, accessed 12 Mar 2012.

[34] Geographical Association. Map of Tectonic Plates. Http://www.geography.org.uk/resources/indonesiantsunami/, accessed 12 Oct 2013.

[35] GeoNet. Earthquake Web Feature Service - Applications and Data - GeoNet, 2013. http://info.geonet.org.nz/display/appdata/Earthquake+Web+Feature+Service, accessed 12 Oct 2013.

[36] J. R. Gibson, H. Lin, and M. A. Bruns. A comparison of fractal analytical methods on 2- and 3-dimensional computed tomographic scans of soil aggregates. *Geoderma*, 134, 2006.

[37] P.A. Gillespie, C.B. Howard, J.J. Walsh, and J. Watterson.

[38] R. Gloaguen, P. R. Marpu, and I. Niemeyer. Automatic extraction of faults and fractal analysis from remote sensing data. *Nonlinear Processes in Geophysics*, 14, 2007.

[39] GNS. GNS_Greendale_fault_trace_from_the_air_Sept 2011, 2011. http://www.gns.cri.nz/Home/Learning/Science-Topics/Earthquakes, accessed 10 Oct 2013.

[40] GNS Science. Plate Collision in NZ. http://www.gns.cri.nz/Home/Learning/Science-Topics/Earthquakes/Earthquakes-at-a-Plate-Boundary/Plate-Collision-in-NZ, accessed 12 Oct 2013.

[41] GNS Science. QMAP seamless digital data 2012. Geological map of New Zealand 1:250,000 (pre-release version), 2012.

[42] D. Gospodinov, E. Marekova, and A. Marinov. Verifying the Dependence of Fractal Coefficients on Different Spatial Distributions. In *7th International Conference of the Balkan Physical Union*, volume 1203, pages 731–736, 2009.

[43] P. Hall and A. Wood. On the performance of box-counting estimators of fractal dimension. *Biometrika*, 80(1):246–252, 1993.

[44] W. K. Hamblin and E. H. Christiansen. *Earth's dynamic systems*. Prentice Hall, 2001.

[45] Holt, Rinehart and Winsto. A Physical Approach - Planet Earth. http://www.mrrclass.net/PhysicalScience/SUPPORTCD/data/0030665787/student/ch17/hk117q.htm, accessed 11 Oct 2013.

[46] S. Hong and D. Huidong. Fractal Dimension Applied in Texture Feature Extraction in X-ray Chest Image Retrieval. In *Proceeding of the IEEE International Conference on Information and Automation*, 2012.

[47] J. Kalda. On the fractality of the biological tree-like structures. *Discrete Dynamics in Nature and Society*, 3(4):297–306, 1999.

[48] N. Kato and X. Lei. Interaction of parallel strike-slip faults and a characteristic distance in the spatial distribution of active faults. *Geophysical Journal International*, 144, 2001.

[49] W. S. Kendal. Evidence for a Fractal Stochastic Process Underlying Measles Epidemics in Britain. *Fractals*, 08(01):29–34, 2000.

[50] W. J. Kious and R. I. Tilling. *This dynamic Earth: the story of plate tectonics*. DIANE Publishing, 1996.

[51] W. J. Kious and R. I. Tilling. *This Dynamic Earth: the Story of Plate Tectonics*. Online Edition, 1996.

[52] A. Kleiner, G. Suchaneck, B. Adolphi, A. Ponomareva, and G. Gerlach. PZT Thin Films Deposited on Copper-Coated Polymer Film Substrates. *Ferroelectrics*, 429, 2012.

[53] S. Kobayashi, T. Maruyama, S. Tsurekawa, and T. Watanabe. Grain boundary engineering based on fractal analysis for control of segregation-induced intergranular brittle fracture in polycrystalline nickel. *Acta Materialia*, 60:6200–6212, 2012.

[54] D. I. Konopinski, S. Hudziak, R.M. Morgan, P.A. Bull, and A.J. Kenyon. Investigation of quartz grain surface textures by atomic force microscopy for forensic analysis. *Forensic Science International*, 223, 2012.

[55] A. N. Kravchenko, A. N. W. Wang, A. J. M. Smucker, and M. L. Rivers. Long-term Differences in Tillage and Land Use Affect Intra-aggregate Pore Heterogeneity. *Soil Science Society of America Journal*, 75(5):1658–1666, 2011.

[56] M. Kuchta. Research at a snail's pace: Fractals in geology, 2009. http://pascals-puppy.blogspot.co.nz/2009/12/fractals-in-geology.html, accessed 07 Nov 2013.

[57] X. Lei and K. Kusunose. Fractal structure and characteristic scale in the distributions of earthquake epicentres, active faults and rivers in Japan. *Geophysical Journal International*, 139:754–762, 1999.

[58] K. Lewis, S. D. Nodder, and L. Carter. Sea floor geology - Active plate boundaries, 2013. http://www.teara.govt.nz/en/map/5586/puysegur-trench, accessed 08 Oct 2013.

[59] LINZ (Crown Copyright Reserved). New Zealand 1: 2 Million. http://ortho.linz.govt.nz/nz_small_scale/nz_2million.jpg, accessed 10 Jan 2014.

[60] B. Mandelbrot. How Long Is the Coast of Britain? Statistical Self-Similarity and Fractional Dimension. *Science*, 156(3775):636–638, 1967.

[61] B. Mandelbrot. *The Fractal Geometry of Nature*. W.H. Freeman, 1977.

[62] B. Mandelbrot. A Theory of Roughness: A talk with Benoit Mandelbrot, 2004. http://www.edge.org/3rd_culture/mandelbrot04/mandelbrot04_index.html, accessed 02 Apr 2013.

[63] Maps Google. Waiheke Island, 2010. https://maps.google.co.nz/, accessed 10 November 2013.

[64] S. Marshak. *Earth: Portrait of a Planet with Geotours Workbook*. W W Norton & Company Incorporated, 3 edition, 2008.

[65] E. Mervine. Monday Geology Picture: Pillow Basalts in the Oman Ophiolite - Georneys - AGU Blogosphere, 2012. http://blogs.agu.org/georneys/2012/01/09/monday-geology-picture-pillow-basalts-in-the-oman-ophiolite/, accessed 19 Oct 2013.

[66] Miguasha National Park. Miguasha: Tectonic context, 2007. www.miguasha.ca/mig-en/tectonic_context.php, accessed 11 Oct 2013.

[67] M. Miller. Close-Up of an Upright Syncline in Limestone, with a Geology Pick for Scale. Montana, USA Photographic Print by Marli Miller at AllPosters.com. http://www.allposters.com/-sp/Close-Up-of-an-Upright-Syncline-in-Limestone-with-a-Geology-Pick-for-Scale-Montana-USA-Posters_i6016238_.htm, accessed 13 Oct 2013.

[68] F. Moisy. boxcount/boxcount.m, 2008. http://www.mathworks.com/matlabcentral/fileexchange/13063-boxcount, accessed 12 Mar 2012.

[69] F. Moisy. boxcount/demo.m, 2008. http://www.mathworks.com/matlabcentral/fileexchange/13063-boxcount/all_files, accessed 12 Mar 2012.

[70] E. M. Moores and R. J. Twiss. *Tectonics*. W. H. Freeman, 1995.

[71] N. Mortimer. New Zealand's Geological Foundations. *Gondwana Research*, 7(1):261–272, 2004.

[72] N.a. Critical Values for the Two-sample Kolmogorov-Smirnov test (2-sided). http://www.soest.hawaii.edu/wessel/courses/gg313/Critical_KS.pdf, accessed 12 Dec 2013.

[73] New World Encyclopedia. Tectonic plates boundaries detailed.

[74] New Zealand GeoNet project and its sponsors EQC and GNS Science and LINZ. Earthquake Resources - Quake Search, 2013. http://info.geonet.org.nz/display/appdata/Earthquake+Resources, accessed 15 Mar 2012.

[75] New Zealand Police. 2011 Christchurch earthquake, 2013. http://www.police.govt.nz/major-events/previous/christchurch-earthquake, accessed 11 Oct 2013.

[76] Oceana. Kermadec–tonga trench. http://oceana.org/en/explore/marine-places/kermadec-tonga-trench, accessed 08 Oct 2013.

[77] C. Oldershaw. The Earth in our hands, 2001.

[78] D. Oldroyd. Classic papers - Arthur Holmes' paper of 1929 on convection currents within the Earth as a cause of. *Episodes*, 34(1):41–50, 2011.

[79] A. J. Pellionisz. Neural Geometry - Towards a Fractal Model of Neurons. In R.J.J. Cotterill, editor, *Models of Brain Function*, pages 453–464. Cambridge University Press, 1989.

[80] Ardmore Pilot. Stories from the Sky: July 2012. http://ardmorepilot.blogspot.co.nz/2012_07_01_archive.html, accessed 06 Nov 2013.

[81] Project Gutenberg. The Notebooks of Leonardo Da Vinci, Complete, 2004. http://www.searchengine.org.uk/pdfs/3/91.pdf, accessed 10 Nov 2013.

[82] V. Radonić, K. Palmer, and Stojanović.

[83] M. Rattenbury. Seamless QMAP, 2012. Mail from 03 April 2012.

[84] R. Reif, J. Qin, L. An, Z. Zhi, S. Dziennis, and R. Wang. Quantifying Optical Microangiography Images Obtained from a Spectral Domain Optical Coherence Tomography System. *International Journal of Biomedical Imaging*, 2012, 2012.

[85] M. Reyners. Why does New Zealand have lots of earthquakes ?, 2013. http://mtaspiring.school.nz/Tephra/WhydoesNZhavelotsofearthquakes.htm, accessed 29 Apr 2013.

[86] L. F. Richardson. The problem of contiguity: an appendix of statistics of deadly quarrels. *General systems yearbook*, 6(13):139–187, 1961.

[87] Schumacher, E. F. Small is Beautiful. *The Radical Humanist*, 37:18–22, 1973.

[88] SGS United States Geological Survey. Earthquake Glossary - active fault. http://earthquake.usgs.gov/learn/glossary/?term=activefaultEarthquakeGlossary-activefault, accessed 14 Oct 2013.

[89] SGS United States Geological Survey. New Zealand Geodetic Datum 1949 (NZGD1949). http://www.linz.govt.nz/geodetic/datums-projections-heights/geodetic-datums/new-zealand-geodetic-datum-1949, accessed 13 Dec 2013.

[90] W. Sierpinski. Sur une courbe cantorienne qui contient une image biunivoque et continue de toute courbe donnée. *Comptes Rendus Hebdomadaires des Seances de l'dcademie des Sciences de Paris*, 162, 1916.

[91] D. Smale. *Sediment Trails in Tectonically Active Islands: Heavy Minerals in Use in New Zealand*, volume 58, chapter 22, pages 569–585. Elsevier, 2007.

[92] N. Smirnov. Table for Estimating the Goodness of Fit of Empirical Distributions. *The Annals of Mathematical Statistics*, 19(2):279–281, 1948.

[93] R. Sridhar and S. Balasubramaniam. Fractal Estimation Using Extended Triangularisation and Box Counting Algorithm for any Geo-Referenced Point Data in GIS. *International Journal of Applied Evolutionary Computation*, 3(3):88–108, 2012.

[94] State of Tasmanina - Biodiversity Conservation Branch: DPIPWE. Geodiversity - Gondwana - The great supercontinent, 2010. `http://www.parks.tas.gov.au/file.aspx?id=6555`, accessed 28 Apr 2013.

[95] R. P. Taylor, A. P. Micolich, and D. Jonas. Fractal analysis of Pollock's drip paintings. *Nature*, 399(6735):422, 1999.

[96] R. P. Taylor, B. Spehar, P. Van Donkelaar, and C. M. Hagerhall. Perceptual and Physiological Responses to Jackson Pollock's Fractals. *Frontiers in human neuroscience*, 5(June):60, 2011.

[97] The University of Waikato - Earth Science. The geological history of New Zealand. `http://sci.waikato.ac.nz/evolution/geologicalHistory.shtml`, accessed 08 Oct 2013.

[98] D. L. Turcotte. Fractals, chaos, self-organized criticality and tectonics. *Terra Nova*, 4, 1992.

[99] UNESCO World Heritage Centre. Khajuraho Group of Monuments, 2013. `http://whc.unesco.org/en/list/240`, accessed 19 Oct 2013.

[100] UNESCO World Heritage Centre. Pont du Gard (Roman Aqueduct), 2013. `http://whc.unesco.org/en/list/344`, accessed 19 Oct 2013.

[101] USGS United States Geological Survey. Inside the Earth [This Dynamic Earth, USGS]. `http://pubs.usgs.gov/gip/dynamic/inside.html`, accessed 20 Oct 2013.

[102] F. Vallianatos, E. Kokinou, and P. Sammonds. Non-Extensive Statistical Physics Approach to Fault Population Distribution. A Case Study from the Southern Hellenic Arc (Central Crete). *Acta Geophysica*, 59(4):770–784, 2011.

[103] H. von Koch. Une méthode géométrique élémentaire pour l'étude de certaines questions de la théorie des courbes planes. *Acta mathematica*, 1906.

[104] vsathletics. Cones. `http://www.vsathletics.com/store/Cones.html`, accessed 24 Apr 2013.

[105] B. Wahl, P. V. Roy, M. Larson, and E. Kampman. *Exploring Fractals on the Macintosh*. Addison-Wesley Professional, 1994.

[106] J. J. Walsh and J. Watterson. Fractal analysis of fracture patterns using the standard box-counting technique: valid and invalid methodologies. *Journal of Structural Geology*, 15(12):1509–1512, 1993.

[107] A. Wegener. *Die Entstehung der Kontinente und Ozeane*. Borntraeger Gebrueder, 2005.

[108] R. Yan and X. Guo. Nonlinear fractal dynamics of human colonic pressure activity based upon the box-counting method. *Computer Methods in Biomechanics and Biomedical Engineering*, pages 1–9, 2012.

[109] C. B. Yang. The origin of power-law distributions in self-organized criticality. *Journal of Physics A: Mathematical and General*, 37(42):523–529, 2004.

[110] H. Zhang and Z. Li. Fractality and Self-Similarity in the Structure of Road Networks. *Annals of the Association of American Geographers*, 102(2):350–365, 2012.

[111] K. Zhang and S. Wang. Determination of the Box-Counting Fractal Dimension of Pore Distribution in Eggshell Based on Scanning Electron Microscopy Image Analysis. *Advanced Materials Research*, 341-342, 2012.

[112] J. Zhao, S. Chen, R. Zuo, and E. J. M. Carranza. Mapping complexity of spatial distribution of faults using fractal and multifractal models: vectoring towards exploration targets. *Computers and Geosciences*, 37(12):1958–1966, 2011.

[113] H. Zhou, X. Peng, S. Peth, and TQ. Xiao. Effects of vegetation restoration on soil aggregate microstructure quantified with synchrotron-based micro-computed tomography. *Soil and Tillage Research*, 124, August 2012.